HEWLETT-PACKARD

HP-67/HP-97

Math Pac I

The program material contained herein is supplied without representation or warranty of any kind. Hewlett-Packard Company therefore assumes no responsibility and shall have no liability, consequential or otherwise, of any kind arising from the use of this program material or any part thereof.

PRINTED IN U.S.A. © HEWLETT-PACKARD COMPANY 1976

Introduction

The 19 programs of Math Pac I have been drawn from the fields of number theory, algebra, trigonometry, analytical geometry, calculus, and special functions.

Each program in this pac is represented by one or more magnetic cards and a section in this manual. The manual provides a description of the program with relevant equations, a set of instructions for using the program, and one or more example problems, each of which includes a list of the actual keystrokes required for its solution. Program listings for all the programs in the pac appear at the back of this manual. Explanatory comments have been incorporated in the listings to facilitate your understanding of the actual working of each program. Thorough study of a commented listing can help you to expand your programming repertoire since interesting techniques can often be found in this way.

On the face of each magnetic card are various mnemonic symbols which provide shorthand instructions to the use of the program. You should first familiarize yourself with a program by running it once or twice while following the complete User Instructions in the manual. Thereafter, the mnemonics on the cards themselves should provide the necessary instructions, including what variables are to be input, which user-definable keys are to be pressed, and what values will be output. A full explanation of the mnemonic symbols for magnetic cards may be found in appendix A.

If you have already worked through a few programs in Standard Pac, you will understand how to load a program and how to interpret the User Instructions form. If these procedures are not clear to you, take a few minutes to review the sections, Loading a Program and Format of User Instructions, in your Standard Pac.

Several programs in this pac were based on programs submitted to the HP-65 Users' Library. We wish to acknowledge the following contributors:

John Joseph Herro for *Optimal Scale for a Graph,*

Rene S. Julian for *Rotations in Three-Dimensional Space,*

Stuart D. Augustin for *Bessel Functions,*

Charles R. Ammerman for *Extended Complementary Error Function.*

We hope that Math Pac I will assist you in the solution of numerous problems in your discipline. We would very much appreciate knowing your reactions to the programs in this pac, and to this end we have provided a questionnaire inside the front cover of this manual. Would you please take a few minutes to give us your comments on these programs? It is in the comments we receive from you that we learn how best to increase the usefulness of programs like these.

CONTENTS

A WORD ABOUT PROGRAM USAGE

This application pac has been designed for both the HP-97 Programmable Printing Calculator and the HP-67 Programmable Pocket Calculator. The most significant difference between the HP-67 and the HP-97 calculators is the printing capability of the HP-97. The two calculators also differ in a few minor ways. The purpose of this section is to discuss the ways that the programs in this pac are affected by the difference in the two machines and to suggest how you can make optimal use of your machine, be it an HP-67 or an HP-97.

Most of the computed results in this pac are output by PRINT statements: most often by the statement PRINTx, and occassionally by the command PRINT STACK. On the HP-97 these results will be output on the printer. On the HP-67 each PRINT command will be interpreted as a PAUSE: the program will halt, display the result for about 5 seconds, then continue execution. The term "PRINT/PAUSE" is used to describe this output condition.

If you own an HP-67, you may want more time to copy down the number displayed by a PRINT/PAUSE. All you need to do is press any key on the keyboard. If the command being executed is PRINTx (eight rapid blinks of the decimal point), pressing a key will cause the program to halt. If the command being executed is PRINT STACK (two slow blinks of the decimal per value), the number in the display will remain there until the depressed key is released; then the next register in the stack will be displayed, and so on. After display of all four registers, the program will halt execution if a key was pressed at any time during the display of the stack contents. In both cases execution of the halted program may be re-initiated by pressing **R/S**.

HP-97 users may also want to keep a permanent record of the values input to a certain program. A convenient way to do this is to set the Print Mode switch to NORMAL before running the program. In this mode all input values and their corresponding user-definable keys will be listed on the printer, thus providing a record of the entire operation of the program.

Several programs in this pac allow you to choose an optional mode which will be referred to on the magnetic card as AUTO. This will apply only to programs that output a long list of results and will allow those results to be output automatically through PRINT/PAUSE commands. If AUTO is not selected, each computed value will be output on the display and the program will halt. The purpose of AUTO mode is to afford maximum convenience to users of both the HP-67 and the HP-97. On the HP-97 it is simplest to have a printed record of each computed result; this can be accomplished just by specifying AUTO. On the HP-67, if every result is to be written down, it may be advantageous *not* to select AUTO, and thus force the program to halt each time a result is found.

Another area that could reflect differences between the HP-67 and the HP-97 is in the keystroke solutions to example problems. It is sometimes necessary in these solutions to include operations that involve prefix keys, namely, **f** on the

HP-97 and [f], [g], and [h] on the HP-67. For example, the operation [10^x] is performed on the HP-97 as [f] [10^x] and on the HP-67 as [g] [10^x]. In such cases, the keystroke solution omits the prefix key and indicates only the operation (as here, [10^x]). As you work through the example problems, take care to press the appropriate prefix keys (if any) for your calculator.

Also in keystroke solutions, those values that are output by the command PRINTx will be followed by three asterisks (***).

FACTORS AND PRIMES

This program will find all prime factors of a positive integer n, or list all prime numbers between lower and upper bounds specified by the user.

A routine under LBL a is used in determining the factors of an integer n. This routine selects a trial divisor d and tests d as a factor of n. If d divides n, then $n \leftarrow n/d$ and d is tested as a factor of the new n. If d does not divide n, a new d is selected. The process continues until $d > \sqrt{n}$, at which point n is returned as the final factor. The trial divisor d takes on the values 2,3,5,7; then for $d > 10$, d takes on those values that satisfy $(d - 10) \bmod 30 = 1, 3, 7, 9, 13, 19, 21,$ or 27. Thus in the first cycle of 30 integers from 11 to 40, d takes on the values 11, 13, 17, 19, 23, 29, 31, and 37. This technique eliminates from the test those values of d ($d>10$) which are divisible by 2, 3, or 5.

To list primes, a lower bound for the search must be specified. The upper bound is an optional input; if omitted, a default value of 2×10^9 is assumed. Upper and lower bounds need not be integers. The search for primes also uses LBL a to determine if an integer n has any factors or is indeed prime. Once an integer n ($n \geq 3$) has been tested and found to be either prime or non-prime, the next integer tested is $n+2$.

Remarks:

1. The number n to be factored must be an integer in the range $0 < n \leq 2 \times 10^9$. Any other input will result in a program halt with a display of "Error".
2. The upper bound of the search for primes must be greater than or equal to the lower bound, or an Error halt will occur.
3. AUTO mode is available to allow automatic output of all calculated results through PRINT/PAUSE commands. The end of all calculations is signalled by a 0.00 in the display for both modes.
4. Either routine can be quite time-consuming for large integers.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
2	To allow automatic output of			
	results, set AUTO mode.		E	1.00
3	To cancel AUTO mode later.		E	0.00

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
4	To find factors, go to step 5;			
	to find primes, go to step 6.			
	FACTORS			
5	Key in the integer and find its			
	prime factors (0.00 signals end).	n	A	Factors
				0.00
	PRIMES			
6	Key in the lower bound of the			
	search for primes.	FROM	B	FROM
7	(optional) Key in the upper			
	bound of the search (if omitted,			
	TO = 2×10^9).	TO	C	TO
8	Find all primes between FROM			
	and TO (0.00 signals end of			
	calculation).		D	Primes
				0.00

Example 1:

Find the prime factors of 924. Do not set AUTO mode.

Keystrokes:	**Outputs:**	
924 A ⟶	2.00	
R/S ⟶	2.00	
R/S ⟶	3.00	
R/S ⟶	7.00	
R/S ⟶	11.00	
R/S ⟶	0.00	(end)

Thus $924 = 2 \times 2 \times 3 \times 7 \times 11$.

Example 2:

Find the prime factors of 3623. Do not use AUTO mode.

Keystrokes:	**Outputs:**	
3623 [A] →	3623.00	
[R/S] →	0.00	(end)

3623 is prime.

Example 3:

Find all prime numbers between 101 and 125. Use AUTO mode.

Keystrokes:	**Outputs:**	
101 [B] 125 [C] [E] →	1.00	(AUTO set)
[D] →	101.00 ***	
	103.00 ***	
	107.00 ***	
	109.00 ***	
	113.00 ***	
	0.00	(end)

GREATEST COMMON DIVISOR, LEAST COMMON MULTIPLE, DECIMAL TO FRACTION

This program contains three different routines: greatest common divisor, least common multiple, and decimal to fraction.

Given integers a and b, the first routine finds their greatest common divisor, GCD(a,b). Optional outputs of this routine are the values of the integers s and t which satisfy the equation GCD(a,b) = sa + tb. The second routine will calculate, for integers a and b, their least common multiple, LCM(a,b). This routine is independent of the first, although both share a common subroutine, LBL e.

The basic algorithm used in finding GCD(a,b) is as follows:

1. If $b = 0$, GCD(a,b) ← a and the program halts.
2. If $b \neq 0$, $z \leftarrow a \bmod b$, $a \leftarrow b$, and $b \leftarrow z$. Return to 1.

The algorithm is actually extended somewhat to allow calculation of s and t. Full details may be found in the reference below.

LCM(a,b) is found by

$$\mathrm{LCM}(a,b) = \frac{ab}{\mathrm{GCD}(a,b)}$$

The third routine in this program will find rational fractional approximations for decimal values by the method of continued fractions. Each successive approximation is closer to the decimal value than the previous one. For example, if the decimal keyed in is 0.33, the first fractional approximation computed will be 1/3. The program will output first the numerator 1, then the denominator 3, then the 10-digit value of the approximation, 0.333333333, and finally the error in this approximation, displayed in scientific notation. The error is found by subtracting the original value, 0.33, from the value of this approximation. At this step the error is 3.333333300-03.

The program will then go on to compute a closer fractional approximation. In this example, the next approximation would be 33/100. Since this is the exact equivalent of the original decimal value, the program will halt after this step displaying 0.000000000. The last fraction generated can be recalled by pressing **D**.

Equations:

The algorithm employed in this routine uses a method of continued fractions, so that the nth fractional approximation f_n is computed as

$$f_n = a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{a_4 + \ddots + \cfrac{1}{a_n}}}} .$$

Each f_i is output as a numerator N_i and a denominator D_i, which are computed as follows:

$$N_i = a_i N_{i-1} + N_{i-2}$$

$$D_i = a_i D_{i-1} + D_{i-2}$$

where $N_{-1} = 0$, $D_{-1} = 1$, $N_0 = 1$, and $D_0 = 0$ by definition.

The values for the a_i may be found by the following algorithm:

Let Dec be the original decimal keyed in. Then $a_1 = \text{INT(Dec)}$. Let $x_1 = 1$ and let $y_1 = \text{FRAC(Dec)}$. Then

$$a_{i+1} = \text{INT}\ (x_i/y_i)$$

$$x_{i+1} = y_i$$

$$y_{i+1} = x_i - a_{i+1} y_i .$$

Remarks:

AUTO mode is available on the Decimal to Fraction routine.

References:

(GCD,LCM) D. E. Knuth, *The Art of Computer Programming,* Vol. 2, Addison-Wesley, 1969.

(Decimal to fraction) Charles G. Moore, *An Introduction to Continued Fractions,* National Council of Teachers of Mathematics, 1964.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
2	For greatest common divisor,			
	go to step 3; for least common			
	multiple, go to step 5; for deci-			
	mal to fraction, go to step 6.			
	GCD			
3	Key in integers a and b and find			
	their greatest common divisor.	a	ENTER↑	
		b	A	GCD (a,b)
4	(optional) Compute coefficients			
	s and t such that GCD (a,b)			
	= sa + tb.		R/S	s
				t
	LCM			
5	Key in integers a and b and find			
	their least common multiple.	a	ENTER↑	
		b	B	LCM (a,b)
	DECIMAL→FRACTION			
6	To allow automatic output of			
	results, set AUTO mode.		E	1.00
7	To cancel AUTO Mode later.		E	0.00
8	Key in a decimal value and find			
	successive fractional approxi-			
	mations (i = 1, 2, ...).	Dec	C	Num_i
				Den_i
				Num_i/Den_i
				$Error_i$

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
9	To re-output last fractional			
	approximation ($Error_n$ shown in			
	display only).		D	Num_n
				Den_n
				Num_n/Den_n
				$Error_n$

Example 1:

Find the greatest common divisor of 406 and 266. Find also the constants s and t.

Keystrokes:	Outputs:
406 ENTER↑ 266 A ⟶	14.00 *** (GCD)
R/S ⟶	2.00 *** (s)
	-3.00 *** (t)

That is, $(2 \times 406) + (-3 \times 266) = 14$.

Example 2:

Find the least common multiple of 406 and 266.

Keystrokes:	Outputs:
406 ENTER↑ 266 B ⟶	7714.00 *** (LCM)

Example 3:

A gear designer wants to reduce the angular speed of a rotating shaft by a factor of 0.45647. He will do this by having a *gear* on the driven shaft mesh with a smaller gear, called a *pinion,* on the drive shaft. If N_g and N_p are the number of teeth on the gear and pinion respectively, then the reduction in speed is by a factor of N_p/N_g. Find the best values for N_p and N_g subject to the constraint that neither value exceed 100. Do not use AUTO mode.

Keystrokes:	Outputs:	
.45647 C ⟶	1.	(Num_1)
R/S ⟶	2.	(Den_1)
R/S ⟶	0.500000000	($Frac_1$)
R/S ⟶	4.353000000–02	($Error_1$)

R/S	→	5.	(Num_2)
R/S	→	11.	(Den_2)
R/S	→	0.454545455	($Frac_2$)
R/S	→	-1.924545500-03	($Error_2$)
R/S	→	21.	(Num_3)
R/S	→	46.	(Den_3)
R/S	→	0.456521739	($Frac_3$)
R/S	→	5.173910000-05	($Error_3$)
R/S	→	173.	($Num_4 > 100$, so stop)

The best values are thus $N_p = 21$, $N_g = 46$.

Example 4:

Generate the series of fractional approximations to π. Use AUTO mode.

Keystrokes:		**Outputs:**	
E	→	1.00	(AUTO set)
π C	→	3. ***	
		1. ***	
		3.000000000 ***	
		-1.415926540-01 ***	
		22. ***	
		7. ***	
		3.142857143 ***	
		1.264489000-03 ***	
		333. ***	
		106. ***	
		3.141509434 ***	
		-8.322000000-05 ***	
		355. ***	
		113. ***	
		3.141592920 ***	
		2.660000000-07 ***	

```
    104348. ***
     33215. ***
3.141592654 ***
0.000000000
```

BASE CONVERSIONS

This program will convert a positive number in base b, x_b, to its equivalent representation in base B, x_B. The bases b and B may take on integer values from 2 to 99, inclusive. Inputs to the program are x_b, b, and B; the single output is the value of x_B. Input of either base, b or B, may be omitted if its value is 10 since a default value of 10 is assigned to both b and B upon input of x_b to key **A**. If several conversions are to be done between the same two bases, i.e., there are several values of x_b for the same b and B, then the bases need not be re-input each time. Once the new value of x_b is keyed in, then pressing **E** will immediately cause the calculation of x_B, based on the most recent values for b and B.

The heart of this program is a routine under LBL e which actually converts numbers to and from base 10 representations. If either b or B is equal to 10, this routine is executed just once, and then the program halts displaying x_B. If, on the other hand, neither b nor B is 10, then x_b is first converted to its decimal representation, x_{10}, and next x_{10} is converted to x_B. Thus the routine is here executed twice.

A number such as $4B6_{16}$ cannot be represented directly on the display because the display is strictly numeric. Therefore, some convention must be adopted to represent numbers R_a when $a > 10$. We use the convention of allocating two digit locations for each single character in R_a when $a > 10$.

For example, $4B6_{16}$ is represented as 041106_{16} by our convention (in hexadecimal system, A = 10, B = 11, C = 12, D = 13, E = 14, F = 15).

When displayed, this number may appear as 41106 or with an exponent

4.1106 04

which is interpreted as $4.B6 \times 16^2$.

The displayed exponent 4 is for base 10 and only serves to locate the decimal point (in the same manner as for decimal numbers).

When base $a > 10$ (as in the above example), divide the displayed exponent by 2 to get the true exponent of the number. When the displayed exponent is an odd integer, shift the decimal point of the displayed number one place (to the left or right) and adjust its exponent accordingly to make the true exponent an integer.

For example, the displayed number

1.112 -03

is interpreted as $B.C \times 16^{-2}$ or $0.BC \times 16^{-1}$.

Remarks:

1. When the magnitude of the number is very large or very small, this program will take a long time to execute.
2. The program will not give any Error indication for invalid inputs for x_b. For example, 981_8 will be treated the same as 1201_8.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
2	To cause input values to be			
	output, set Print/Pause mode.		f A	1.00
3	To cancel Print/Pause mode.		f A	0.00
4	Key in number in first base.	x_b	A	
5	(optional) Key in first base.	b	B	
	(If omitted, default value of b			
	is 10.)			
6	(optional) Key in second base	B	C	
	(If omitted, default value of B			
	is 10.)			
7	Calculate number in second			
	base.		D	x_B
8	To convert another number			
	between the *same* two bases			
	(from b to B), key in the new			
	x_b and find the new x_B.	x_b	E	x_B
9	To change either base, go to			
	step 4.			

Example 1:

The following octal numbers (b = 8) are addresses of a segment of a program in an HP2100 minicomputer: 177700, 177735, 177777. What are the values of these addresses in base 10 (B = 10)?

Keystrokes:	Outputs:
177700 [A] 8 [B] [D] ——→	65472.00 ***
177735 [E] ——→	65501.00 ***
177777 [E] ——→	65535.00 ***

Example 2:

Find the ten-digit binary representation of π. (x_b = 3.141592654, b = 10, B = 2)

Keystrokes:	Outputs:
[π] [A] 2 [C] [D] [DSP] [9] ——→	11.00100100

Example 3:

Convert the following octal numbers (b = 8) into hexadecimal (B = 16): 7.200067×8^{-10}, $1.513561778 \times 8^{17}$

Keystrokes:	Outputs:
7.200067 [EEX] [CHS] 10 [A] 8 [B]	
16 [C] [D] ——→	1.130000031−14 ***
	($1.D003A \times 16^{-7}$)
1.513561778 [EEX] 17 [E] ——→	1.302141404 25 ***
	(13.02141404 24
	$=D.2EE4 \times 16^{12}$)

OPTIMAL SCALE FOR A GRAPH; PLOTTING

Two separate routines are included in this program. The first finds the optimal scale for a graph, given certain parameters of the graph. The second routine is designed to be of assistance in plotting functions of one variable by generating ordered pairs (x,f(x)) for a range of x-values specified by the user.

Optimal scale for a graph

In the first routine the input parameters are the minimum and maximum values on the graph (Min and Max) and the number of major divisions (tics) from top to bottom of the graph. The routine will select a "nice" scale for the graph, meaning that the graph will fill as much of the page as possible, subject to these constraints: (1) the quantity Δ represented by one major division will be 1, 2, 4 or 5 times an integral power of 10; (2) the bottom and the top of the graph will be integral multiples of one division; and (3) bottom $\leqslant$ Min and top $\geqslant$ Max. Outputs of the routine are values for the top and bottom of the graph; the amount of each major division, Δ; and the "efficiency," or percentage of the page filled by the graph. Efficiency is found by [(Max − Min)/(Top − Bot)] x 100.

Plotting

In the second routine, the function f(x) must be specified and loaded into program memory by the user. The user must also input beginning and ending values for x (Beginx and Endx), and the step size or increment used for x (Step). Then the routine will output the values of (x,f(x)) for the successive values of x represented by

$$x_j = \text{Beginx} + j\text{Step} \quad , \quad j=0, 1,2,\ldots,n$$

where n is such that $x_{n+1} > \text{Endx}$. The end of calculations is signalled by a 0.00 in the display.

The AUTO option is provided for output of the ordered pairs (x,f(x)) through Print/Pause commands. If AUTO is not selected, the values will be output one at a time by the use of R/S.

Although we have discussed only one f(x), there may actually be up to five different functions $f_i(x)$, $i=1,2,\ldots,5$, in program memory at one time. Each function should be under its own label, 1 through 5, and should be followed by

RTN. The function to be evaluated is specified by keying in 1, 2, 3, 4, or 5 and pressing [f] [E].

92 program steps are available to the user for specifying functions $f_i(x)$. This includes all LBL and RTN statements. The functions should assume that upon entry the value of x will be found in the x-register. Registers R_0 through R_9 and R_{S0} through R_{S9}, as well as the stack, are available to the user. The functions $f_i(x)$ may use up to two levels of subroutines: note, however, that the only unused labels are 1 through 5.

To specify your functions, you may wish to record them on a blank magnetic card for rapid entry. Alternatively, you may key them into program memory after loading side 1 and side 2 of this card. To link in recorded functions, follow these steps:

1. Load side 1 and side 2 of *Optimal Scale, Plotting*.
2. Press [GTO] [•] [1] [3] [2].
3. Press [MERGE].
4. Load your own magnetic card with the functions $f_i(x)$ recorded.

To key in a new function:

1. Load side 1 and side 2 of *Optimal Scale, Plotting*.
2. Press [GTO] [•] [1] [3] [2].
3. Key in your function(s), beginning each with LBL (1 through 5) and ending each with RTN.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
2	For optimal scale of a graph,			
	go to step 3; for plotting, go to			
	step 7.			
	OPTIMAL SCALE FOR			
	A GRAPH			
3	Key in the minimum value on			
	the graph.	Min	A	Min
4	Key in the maximum value on			
	the graph.	Max	B	Max
5	Key in the number of tics desired			
	and find the graph top, bottom,			
	value of one tic, and % efficiency.	Tics	C	Top
				Bottom
				Δ
				%
6	To change any value, go to the			
	appropriate step, then to step 5.			
	PLOTTING			
7	Load subroutine(s) (either key			
	them in with LBL and RTN,			
	or link from step 132).			
8	Select function under LBL 1, 2,			
	3, 4 or 5.	i (1–5)	f E	i
9	Key in the beginning x-value.	Begin x	f A	Begin x
10	Key in the final or maximum			
	x-value.	End x	f B	End x
11	Key in the step size for x.	Step	f C	Step
12	For automatic output of results,			
	go to step 13; for manual output,			
	go to step 16.			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
	AUTO mode			
13	Set AUTO mode to allow auto-			
	matic output of results.		E	1.00
14	To cancel AUTO mode later		E	0.00
15	Output successive ordered			
	pairs; program will halt displaying			
	0.00 when x > End x.		f D	x
				f_i (x)
	Manual mode			
16	Output first ordered pair.		f D	x
			R/S	f_i (x)
17	For all successive ordered pairs;			
	0.00 signals end (x > End x).		R/S	x
			R/S	f_i (x)
18	The value for i may be changed			
	at any time. Begin x, End x, and			
	Step need not be re-input if			
	their values are unchanged.			

Example 1:

Find the best scale to graph a function whose minimum is 20, maximum is 40, with 5 major divisions from top to bottom (figure 1).

Keystrokes:	Outputs:	
20 [A] 40 [B] 5 [C] ⟶	40.00	(Top)
	20.00	(Bottom)
	4.00	(Δ)
	100.00	(% Efficiency)

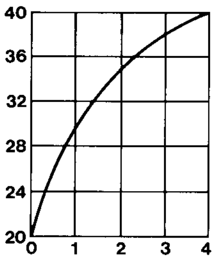


Figure 1

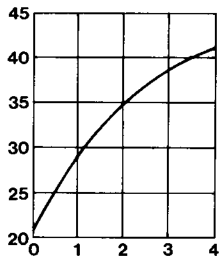


Figure 2

Example 2:

Suppose the minimum changes to 21, the maximum to 41, with the number of tics still 5. Find the new optimal scale (figure 2).

Keystrokes:	Outputs:	
21 [A] 41 [B] 5 [C] ⟶	45.00	(Top)
	20.00	(Bottom)
	5.00	(Δ)
	80.00	(% Efficiency)

Example 3:

Two different functions are to be plotted in the range from 2.00 to 3.00. The first is $f_1(x) = e^x$, and the second is $f_2(x) = x^x$. Use a step size of 0.25 for $f_1(x)$ and 0.2 for $f_2(x)$. Load $f_1(x)$ under LBL 1 and $f_2(x)$ under LBL 2. Use AUTO mode with $f_2(x)$.

Keystrokes: **Outputs:**

[GTO] [·] [1] [3] [2]

Switch to PRGM.

[LBL] [1] [e^x] [RTN] [LBL] [2]

ENTER↑ y^x RTN

Switch to RUN.

1 f E 2 f A 3 f B

.25 f C f D →	2.00	(x)
R/S →	7.39	(e^x)
R/S →	2.25	
R/S →	9.49	
R/S →	2.50	
R/S →	12.18	
R/S →	2.75	
R/S →	15.64	
R/S →	3.00	
R/S →	20.09	
R/S →	0.00	(end)
2 f E .2 f C E →	1.00	(AUTO set)
f D →	2.00 ***	(x)
	4.00 ***	(x^x)
	2.20 ***	
	5.67 ***	
	2.40 ***	
	8.18 ***	
	2.60 ***	
	11.99 ***	
	2.80 ***	
	17.87 ***	
	3.00 ***	
	27.00 ***	
	0.00	(end)

COMPLEX OPERATIONS

This program allows for chained calculations involving complex numbers. The four operations of complex arithmetic (+, −, ×, ÷) are provided, as well as several of the most used functions of a complex variable z ($|z|$, $1/z$, z^n, $z^{1/n}$, and e^z). Functions and operations may be mixed in the course of a calculation to allow evaluation of expressions like $z_3/(z_1 + z_2)$, $e^{z_1 z_2}$, $|z_1 + z_2| + |z_2 - z_3|$, etc., where z_1, z_2, z_3 are complex numbers of the form a + ib.

Keying in a complex number

A complex number is input to the program by keying in its real part, pressing [ENTER↑], keying in its imaginary part, and pressing [A]. For example, the complex number $z_1 = 2 + 3i$ is input as 2 [ENTER↑] 3 [A]. This number is then stored by the program. There is room in the program to remember up to two complex numbers at a time. A second complex number $z_2 = 5 - i$ could be input as 5 [ENTER↑] 1 [CHS] [A]. The program would now contain both the first and the second complex number.

Functions

The complex functions in this program act on just one number. Thus to perform a function, you need simply to input a complex number z and then perform the appropriate function. For example, to find the reciprocal of (2 + 3i), press 2 [ENTER↑] 3 [A] [f] [B]. The result is calculated as a + ib = 0.15 − 0.23 i. This result is now stored in place of the original number, and further calculations will operate on this result. All complex functions operate in this same manner, with one exception: since the function $|z|$ returns a real number, its result is not stored.

Arithmetic Operations

An arithmetic operation needs two numbers to operate on. Both numbers must be input before the operation can be performed. Suppose that $z_1 = 2 + 3i$, $z_2 = 5 - i$, and we wish to find $z_1 - z_2$. This can be calculated by the keystrokes 2 [ENTER↑] 3 [A] 5 [ENTER↑] 1 [CHS] [A] [C]. The result $z_3 = a + ib$ is found to be −3 + 4i. This result is now stored by the program in place of the *second* complex number z_2. A further calculation $z_3 \times z_4$ could be performed by inputting z_4 and pressing [D] for multiplication. This type of chaining can be continued indefinitely, and functions can be interspersed with arithmetic operations.

Equations:

Let $z_k = a_k + ib_k = r_k e^{i\theta_k}$, $k = 1, 2$

$$z = a + ib = re^{i\theta}$$

Let the result in each case be u + iv.

$$z_1 + z_2 = (a_1 + a_2) + i(b_1 + b_2)$$

$$z_1 - z_2 = (a_1 - a_2) + i(b_1 - b_2)$$

$$z_1 z_2 = r_1 r_2 \, e^{i(\theta_1 + \theta_2)}$$

$$z_1/z_2 = \frac{r_1}{r_2} \, e^{i(\theta_1 - \theta_2)}$$

$$|z| = \sqrt{a^2 + b^2}$$

$$1/z = \frac{a}{r^2} - i\frac{b}{r^2}$$

$$z^n = r^n \, e^{in\theta}$$

$$z^{1/n} = r^{1/n} \, e^{i\left(\frac{\theta}{n} + \frac{360k}{n}\right)}, \; k=0,1, \ldots, n-1$$

(All n roots will be output and temporarily stored, $k = 0, 1, \ldots, n-1$; at the end of the calculation, the final root will be stored.)

$e^z = e^a (\cos b + i \sin b)$, where b is in radians.

Remarks:

The logic system for this program may be thought of as a kind of Reverse Polish Notation (RPN) with a stack whose capacity is two complex numbers. Let the bottom register of the complex stack be ξ and the top register τ. These are analogous to the X- and T-registers in the calculator's own four-register stack.* A complex number z_1 is input to the ξ-register by the keystrokes a_1 ENTER↑ b_1 A. Upon input of a second complex number z_2 (as a_2 ENTER↑ b_2 A), z_1 is moved to τ and z_2 is placed in ξ. The previous contents of τ are lost.

*Each register of the complex stack must actually hold two real numbers: the real and the imaginary part of its complex contents. Thus it takes two of the calculator's registers to represent one register in the complex stack. We will speak of the complex stack registers as though they were each just one register.

Functions operate on the ξ-register, and the result (except for $|z|$) is left in ξ; τ is unchanged. Arithmetic operations involve both the ξ- and τ-registers; the result of the operation is left in ξ and τ is unchanged.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
2	Key in first complex number			
	(a_1 + i b_1).	a_1	ENTER↑	
		b_1	A	
3	For a function, go to step 7; for			
	arithmetic, go to step 4. A com-			
	plex result is u + iv.			
	ARITHMETIC			
4	Key in second complex number			
	(a_2 + i b_2).	a_2	ENTER↑	
		b_2	A	
5	Select one of four operations:			
	• Add (+)		B	u
				v
	• Subtract (–)		C	u
				v
	• Multiply (x)		D	u
				v
	• Divide (÷)		E	u
				v
6	The result of the operation has			
	been stored; go to step 7 for a			
	function or to step 4 for further			
	arithmetic.			
	FUNCTIONS			
7	Select one of five functions:			
	• Magnitude ($\|z\|$)		f A	$\|z\|$
	• Reciprocal (1/z)		f B	u

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
				v
	• Raise z to an integer power (z^n)	n	f C	u
				v
	• Find the n^{th} root of z ($z^{1/n}$)			
	Note: n roots (u + iv) will be			
	found.	n	f D	u
				v
	• Raise e to the power z (e^z)		f E	u
				v
8	The result, if complex, has been			
	stored; go to step 4 for arithmetic			
	or to step 7 for another function.			

Example 1:

Evaluate the expression

$$\frac{z_1}{z_2 + z_3},$$

where $z_1 = 23 + 13i$, $z_2 = -2 + i$, $z_3 = 4 - 3i$. (Suggestion: since the program can remember only two numbers at a time, perform the calculation as

$$z_1 \times \left[1/(z_2 + z_3)\right].)$$

Keystrokes:	**Outputs:**
2 CHS ENTER↑ 1 A 4 ENTER↑ 3	
CHS A B ⟶	2.00 *** real ($z_2 + z_3$)
	-2.00 *** imag ($z_2 + z_3$)
f B ⟶	0.25 *** $1/(z_2 + z_3)$
	0.25 ***
23 ENTER↑ 13 A D ⟶	2.50 *** ($z_1/(z_2 + z_3)$)
	9.00 ***

Example 2:

Find the 3 cube roots of 8.

Keystrokes:	**Outputs:**
8 ENTER↑ 0 A 3 f D ⟶	2.00 ***
	0.00 ***
	-1.00 ***
	1.73 ***
	-1.00 ***
	-1.73 ***

Example 3:

Evaluate $e^{z^{-2}}$, where z = (1 + i).

Keystrokes:	**Outputs:**
1 ENTER↑ 1 A 2 f C ⟶	0.00 *** (z^2)
	2.00 ***
f B ⟶	0.00 *** (z^{-2})
	-0.50 ***
f E ⟶	0.88 *** ($e^{z^{-2}}$)
	−0.48 ***

POLYNOMIAL SOLUTIONS

This program will solve polynomial equations with real coefficients of degree 5 and below, provided the high-order coefficient is 1. The equation may be represented as

$$x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0 = 0 \quad , \quad n=2, 3, 4, \text{ or } 5.$$

If the leading coefficient is not 1, it should be made 1 by dividing the entire equation by that coefficient.

The user must store the coefficients of the equation beforehand, beginning with a_0 in R_0 through a_{n-1} in R_{n-1}. Zero must be input for those coefficients which are zero. It is not necessary to store the leading coefficient as 1, or any a_k where $k > n$.

After the coefficients have been stored, the user-definable key ([A] through [D]) which represents the order of the polynomial should be pressed. All roots of the equation, real and complex, will then be computed. For example, if coefficients a_0, a_1, a_2, and a_3 have been stored in registers R_0 through R_3, then key [B] should be pressed to compute the four roots of the fourth degree polynomial equation

$$x^4 + a_3x^3 + a_2x^2 + a_1x + a_0 = 0.$$

Equations:

The routines for third and fifth degree equations use an iterative routine under LBL a to find one real root of the equation. This routine requires that the constant term a_0 not be zero for these equations. (If $a_0 = 0$, then zero is a real root and by factoring out x, the equation may be reduced by one order.) After one root is found, synthetic division is performed to reduce the original equation to a second or fourth degree equation.

To solve a fourth degree equation, it is first necessary to solve the cubic equation

$$y^3 + b_2y^2 + b_1y + b_0 = 0$$

where $b_2 = -a_2$

$b_1 = a_3a_1 - 4a_0$

$b_0 = a_0 (4a_2 - a_3^2) - a_1^2.$

Let y_0 be the largest real root of the above cubic.

Then the fourth degree equation is reduced to two quadratic equations:

$$x^2 + (A + C)x + (B + D) = 0$$
$$x^2 + (A - C)x + (B - D) = 0$$

where $A = \frac{a_3}{2}$, $B = \frac{y_0}{2}$

$$D = \sqrt{B^2 - a_0}$$

$$C = \begin{cases} \left(AB - \frac{a_1}{2}\right) / D & \text{if } D \neq 0 \\ \sqrt{A^2 - a_2 + y_0} & \text{if } D = 0 \end{cases}$$

Roots of the fourth degree equation are found by solving the two quadratic equations.

A quadratic equation $x^2 + a_1x + a_0 = 0$ is solved by the formula $x_{1,2} = -\frac{a_1}{2} \pm \sqrt{\frac{a_1^2}{4} - a_0}$. If $D = \frac{a_1^2}{4} - a_0 > 0$, the roots are real; if $D < 0$, the roots are complex, being $u \pm iv = -\frac{a_1}{2} \pm i\sqrt{-D}$.

A real root is output as a single number. Complex roots always occur in pairs of the form $u \pm iv$. They are output by loading the stack with u, v, u, and $-v$ in registers T, Z, Y, and X, respectively, and then executing the command Print Stack. If these roots are being output through a Pause (HP-67) rather than a Print (HP-97), some attention may be required to make sure that no roots go unnoticed.

Remarks:

1. Long execution times (~1–2 minutes) may be expected for equations of degree 3, 4, or 5, as these use an iterative routine once or more.
2. There is one condition in the solution of fourth or fifth degree polynomials that can cause the program to halt displaying Error. It is a very rare condition and you may never encounter it. It will occur when $b_0 = a_0(4a_2 - a_3^2) - a_1^2 = 0$ in the solution of the cubic to find y_0. If the calculator halts at line 161 displaying Error, then b_0 has been found to be zero and the following key sequence should be performed to recover from the error: 0 STO 7 RCL 1 STO 0 RCL 2 STO 1 D. After execution of D, press GTO . 044 R/S. The program will now continue to execute normally.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
2	Input coefficients below order			
	of polynomial (i.e., for degree n,			
	input through a_{n-1}).			
	Coefficients = 0 must be so			
	input.	a_0	STO 0	
		a_1	STO 1	
		a_2	STO 2	
		a_3	STO 3	
		a_4	STO 4	
3	Compute roots to polynomial			
	of degree			
	• 5		A	Roots 1–5
	• 4		B	Roots 1–4
	• 3		C	Roots 1–3
	• 2		D	Roots 1–2
4	A single number will be output			
	for a real root; complex pairs of			
	roots (u ± iv) will output as			
	shown:			u
				v
				u
				–v
5	For a new equation, return to			
	step 2.			

Example 1:

Solve $x^5 - x^4 - 101x^3 + 101x^2 + 100x - 100 = 0$.

Keystrokes: **Outputs:**

100 CHS STO 0 100 STO 1
101 STO 2 101 CHS STO 3

1 CHS STO 4 A ⟶	10.00 *** (Root 1)
	1.00 *** (Root 2)
	1.00 *** (Root 3)
	−1.00 *** (Root 4)
	−10.00 *** (Root 5)

Example 2:

Solve $4x^4 - 8x^3 - 13x^2 - 10x + 22 = 0$.

Rewrite the equation as $x^4 - 2x^3 - \frac{13}{4}x^2 - \frac{10}{4}x + \frac{22}{4} = 0$.

Keystrokes:	Outputs:	
22 ENTER↑ 4 ÷ STO 0 10 ENTER↑		
4 ÷ CHS STO 1 13 ENTER↑ 4 ÷		
CHS STO 2 2 CHS STO 3 B ⟶	−1.00	(Roots 1 & 2
	1.00	are
	−1.00	−1.00 ± 1.00i)
	−1.00	
	3.12 ***	(Root 3)
	0.88 ***	(Root 4)

Example 3:

Solve $x^3 - 4x^2 + 8x - 8 = 0$.

Keystrokes:	Outputs:	
8 CHS STO 0 8 STO 1		
4 CHS STO 2 C ⟶	2.00 ***	(Root 1)
	1.00	(Roots 2 & 3
	1.73	are
	1.00	1.00 ± 1.73i)
	−1.73	

Example 4:

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

Rewrite the equation as $x^2 + 2.5x + 1.5 = 0$.

Keystrokes:	Outputs:	
1.5 STO 0 2.5 STO 1 D ⟶	−1.00 ***	(Root 1)
	−1.50 ***	(Root 2)

4x4 MATRIX OPERATIONS

This two-card program allows several of the most important operations involving 4x4 matrices, namely, the calculations of the determinant and inverse of a 4x4 matrix, and the solution of a system of simultaneous equations in 4 unknowns.

The method used in this program is that of Gaussian elimination with partial pivoting. Space does not allow a full treatment of the pertinent equations; however, the Comments section of the program listing shows the operations in detail, step by step.

Basically, the first of these two cards, 4x4 Matrix Setup, allows for input of the matrix A and transforms A into an upper triangular matrix U, assuming A is nonsingular. The multipliers used to accomplish this transformation form a lower triangular matrix, L, which has 1's along its diagonal. If we disregard pivoting, a technique of row interchanges which may improve accuracy and which may introduce one or more permutation matrices, then the relationship among these matrices is U = LA. At the end of execution of the first card, the original matrix A no longer exists in memory. The initial elements a_{ij} have been replaced by the elements of U ($i \leq j$) and of L ($i > j$). (The elements of U will still be referred to as a_{ij}; those of L will be called m_{ij} in the program listing comments). The second card, 4x4 Matrix Solutions, uses the transformed matrices U and L to compute the determinant and inverse of A, and to solve systems of simultaneous equations.

Equations:

$$\text{Let } A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}$$

The determinant of A, Det A, is found *after* its transformation to U by the product of the diagonal elements:

$$\text{Det A} = (-1)^k\, a_{11}\, a_{22}\, a_{33}\, a_{44},$$

where k is the number of row interchanges required by pivoting.

A set of 4 simultaneous equations in 4 unknowns may be written as

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4 = b_1$$

$$a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 = b_2$$

$$a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + a_{34}x_4 = b_3$$

$$a_{41}x_1 + a_{42}x_2 + a_{43}x_3 + a_{44}x_4 = b_4,$$

where the $\{x_i\}$ are unknowns and the $\{b_i\}$ constants.

In matrix notation, this becomes A **x** = **b**, where **x** and **b** are the column vectors $\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$ and $\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$ respectively.

This problem is solved (neglecting pivoting) as U**x** = L**b**.

Let C be the inverse of A, i.e., the 4x4 matrix such that AC = CA = I, where I is the 4x4 matrix such that

$$I_{ij} = \begin{cases} 1, & i=j \\ 0, & i \neq j \end{cases} \quad , i, j = 1, 2, 3, 4.$$

C is computed a column at a time in the following way:
let $\mathbf{c}^{(j)}$ be the j^{th} column vector of C, i.e.,

$$\mathbf{c}^{(j)} = \begin{bmatrix} c_{1j} \\ c_{2j} \\ c_{3j} \\ c_{4j} \end{bmatrix} \quad , j = 1, 2, 3, 4.$$

Then $\mathbf{c}^{(j)}$ is found by the solution of the equation

$$A\mathbf{c}^{(j)} = \mathbf{I}^{(j)} \qquad \text{where } \mathbf{I}^{(j)} = \begin{cases} 1 & i=j \\ 0 & i \neq j \end{cases}, i = 1, 2, 3, 4.$$

For example, $\mathbf{c}^{(1)}$ is found by solution of

$$A\,\mathbf{c}^{(1)} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Remarks:

1. A halt during the execution of card 1 (Setup) with a display of ''Error'' indicates that the matrix A is singular.

2. If operations are to be carried out on the same matrix over a period of time, it might be convenient to record the elements of the matrix on a magnetic card for rapid input at a later date. Because the program immediately starts operating on the matrix after the last element has been keyed in, the program needs to be modified to halt after the input of a_{44}. This may be accomplished by the following steps:

    a. Load side 1 and side 2 of 4×4 Matrix Setup.
    b. Press GTO [•] 025.
    c. Switch to PRGM, press [DEL], R/S.
    d. Switch to RUN and press A to start data input.
    e. After the input of a_{44}, the program will halt. At this point, the data may be recorded for later use.
    f. To continue execution, press B.

References:

George E. Forsythe, Michael A. Malcolm, and Cleve B. Moler, *Computer Methods in Mathematical Computation,* Computer Science Department, Stanford University, 1972.

G. Forsythe and C. Moler, *Computer Solution of Linear Algebraic Systems,* Prentice-Hall, 1967.

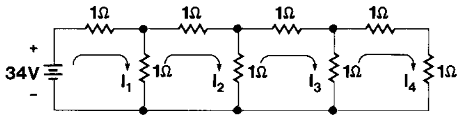
C. Moler, ''Matrix Computations with Fortran and Paging,'' Comm. ACM, vol. 15, no. 4, pp. 268-270 (April, 1972).

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2 of 4 x 4			
	Matix Setup.			
2	If data has already been stored			
	on magnetic card, go to step 7;			
	to key in data, go to step 3.			
3	To cause output of elements			
	$\{a_{ij}\}$ of matrix as they are			
	keyed in, set flag 0.		SF 0	
4	Prepare to input elements of			
	matrix in *column* order (a_{11},			
	a_{21}, a_{31}, a_{41}, a_{12}, a_{22}, etc.)		A	1.1
5	Display shows i.j; key in element			
	in row i, column j.	a_{ij}	R/S	next i. j
6	Repeat step 5 until all elements			
	of matrix have been keyed in;			
	after a_{44} has been keyed in,			
	program execution will begin			
	immediately. Go to step 9.			
7	If matrix data is already stored on			
	magnetic card, load side 1 and			
	side 2 of data card.			
8	Begin program execution.		B	
9	Load side 1 and side 2 of 4 x 4			
	Matrix Solutions.			
10	For automatic output of results,			
	set AUTO mode.		E	1.00
11	To cancel AUTO mode later		E	0.00
12	(optional) Compute determinant.		A	Det A

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
13	To solve a system of four			
	simultaneous equations, key in			
	right-hand side and find **x**.	b_1	ENTER↑	
		b_2	ENTER↑	
		b_3	ENTER↑	
		b_4	B	x_1
				x_2
				x_3
				x_4
14	Find the inverse of matrix A			
	($C = A^{-1}$), displayed in column			
	order.		C	c_{11}
				c_{21}
				c_{31}
				c_{41}
				c_{12}
				c_{22}
				c_{32}
				c_{42}
				c_{13}
				c_{23}
				c_{33}
				c_{43}
				c_{14}
				c_{24}
				c_{34}
				c_{44}
				0.00

Example 1:

By applying the technique of loop currents to the circuit below, find the currents I_1, I_2, I_3, and I_4. Do not use AUTO mode.



The equations to be solved are

$$\begin{array}{ccccccc} 2I_1 & -I_2 & & & = & 34 \\ -I_1 & +3I_2 & -I_3 & & = & 0 \\ & -I_2 & +3I_3 & -I_4 & = & 0 \\ & & -I_3 & +3I_4 & = & 0 \end{array}$$

In matrix form,

$$\begin{bmatrix} 2 & -1 & 0 & 0 \\ -1 & 3 & -1 & 0 \\ 0 & -1 & 3 & -1 \\ 0 & 0 & -1 & 3 \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \\ I_3 \\ I_4 \end{bmatrix} = \begin{bmatrix} 34 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Load side 1 and side 2 of 4x4 Matrix Setup. Prepare to store matrix data on a magnetic card.

Keystrokes:	Outputs:
GTO [•] [0] [2] [5]	
Switch to PRGM	
[DEL] R/S	
Switch to RUN	
A 2 R/S 1 CHS R/S 0 R/S 0 R/S	
1 CHS R/S 3 R/S 1 CHS R/S	
0 R/S 0 R/S 1 CHS R/S 3 R/S	
1 CHS R/S 0 R/S 0 R/S 1 CHS R/S	
3 R/S ⟶	4.0

Program halts, displaying 4.0.

W/DATA → "Crd"

Insert side 1 of a blank magnetic card, see "Crd" and insert side 2.

B → 2.6

Load side 1 and side 2 of 4x4 Matrix Solutions. → 2.62

34 ENTER↑ 0 ENTER↑
0 ENTER↑ 0 B → 21.00 (I_1)
R/S → 8.00 (I_2)
R/S → 3.00 (I_3)
R/S → 1.00 (I_4)

Example 2:

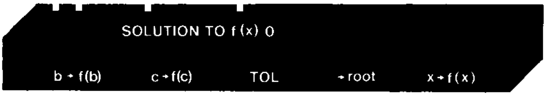
Find the determinant and inverse of the matrix below. Use AUTO mode.

$$\begin{bmatrix} 7 & 5 & 1 & 3 \\ 5 & 7 & 7 & 7 \\ 3 & 3 & 3 & 5 \\ 1 & 1 & 5 & 1 \end{bmatrix}$$

Keystrokes:	Outputs:	
Load side 1 and side 2 of 4x4 Matrix Setup		
A 7 R/S 5 R/S 3 R/S 1 R/S 5 R/S 7 R/S 3 R/S 1 R/S 1 R/S 7 R/S 3 R/S 5 R/S 3 R/S 7 R/S 5 R/S 1 R/S →	2.5	
Load side 1 and side 2 of 4x4 Matrix Solutions →	2.46	
E →	1.00	(AUTO set)
A →	-256.00	(Det A)
DSP 6 C →	0.218750 ***	(c_{11})
	-0.046875 ***	(c_{21})
	-0.015625 ***	(c_{31})
	-0.093750 ***	(c_{41})

−0.281250	***	(c_{12})
0.453125	***	(c_{22})
−0.015625	***	(c_{32})
−0.093750	***	(c_{42})
0.218750	***	(c_{13})
−0.546875	***	(c_{23})
−0.015625	***	(c_{33})
0.406250	***	(c_{43})
0.218750	***	(c_{14})
−0.296875	***	(c_{24})
0.234375	***	(c_{34})
−0.093750	***	(c_{44})
0.000000		(End)

SOLUTION TO f(x)=0 ON AN INTERVAL



This program finds one real root of the equation f(x) = 0 in a finite interval [b,c], where f(x) is a function specified by the user which must be continuous and real on the interval. The program assumes without checking that of the values f(b) and f(c), one will be positive and one negative, i.e., $f(b) \times f(c) < 0$. In this way, b and c will bracket the root. An accuracy tolerance TOL (≥ 0) must also be specified. This number should be the greatest allowable error in the final approximation for the root. That is, the actual root will be no farther away than TOL from the program's solution for the root.

The function f(x) should be keyed into program memory under LBL E and should assume that x will be in the X-register upon entry. 85 program steps, registers R_1 through R_7, R_{S0} through R_{S9}, and the stack are available for defining f(x).

The method used is a combination of bisection (interval-halving) and the secant method. Bisection is often slow but is guaranteed to converge to a root, if one exists in the interval; the secant method is fast but does not always converge. The algorithm employed in this program combines the safety of bisection with some of the speed of the secant method. If the function is known to be well-behaved on the interval in question, then the program in Standard Pac, *Calculus and Roots of f(x),* may lead to a faster and more convenient solution.

Remarks:

As each value for b or c is input, its function value will be computed and displayed. If you are in doubt about values for b and c which will satisfy $f(b) \times f(c) < 0$, you may simply keep inputting different values until you strike a good combination. Each new value input overwrites the old.

References:

George E. Forsythe, Michael A. Malcolm, and Cleve B. Moler, *Computer Methods in Mathematical Computation,* Computer Science Department, Stanford University, 1972.

Richard P. Brent, *Algorithms for Minimization without Derivatives*, Prentice-Hall, 1973.

T. J. Dekker, "Finding a zero by means of successive linear interpolation," in B. Dejon and P. Henrici (editors), *Constructive Aspects of the Fundamental Theorem of Algebra,* Interscience, 1969.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
2	Prepare to key in function.		GTO E	
3	Switch to PRGM. See line 138.			
4	Key in the function f(x) (need			
	not add RTN).			
5	Switch to RUN.			
6	Key in the end points of the			
	interval (remember f(b) x f(c) < 0).	b	A	f(b)
		c	B	f(c)
7	Key in the accuracy tolerance.	TOL	C	TOL
8	Compute a real root.		D	root
9	To evaluate the function at			
	any point.	x	E	f(x)

Example 1:

Find an angle α between 100 and 101 radians such that $\sin \alpha = 0.1$. Hence let $f(x) = \sin x - 0.1$. Assume a tolerance of 10^{-3}.

Keystrokes: **Outputs:**

Load side 1 and side 2.

GTO E

Switch to PRGM. See line 138.

RAD SIN .1 − DEG

Switch to RUN.

Keystrokes		Outputs	
100 A	→	−0.61	(f(100))
101 B	→	0.35	(f(101))
EEX CHS 3 C	→	1.000000000−03	
D	→	100.63	(root)

Example 2:

Find a root of the equation $\ln x + 3x - 10.8074 = 0$ in the interval $[1, 5]$. An accuracy of 10^{-4} is acceptable. Store the constant 10.8074 in R_1.

Keystrokes: **Outputs:**

Load side 1 and side 2.

GTO E

Switch to PRGM. See line 138.

LN LAST X 3 × + RCL 1 −

Switch to RUN.

10.8074 STO 1 →	10.81	
1 A →	-7.81	(f(1))
5 B →	5.80	(f(5))
EEX CHS 4 C →	1.000000000–04	
D →	3.21	(root)

Check the solution by computing its function value.

E →	−1.901000000–05	(f(3.21))

NUMERICAL INTEGRATION

This program will perform numerical integration whether a function is known explicitly or only at a finite number of equally spaced points (discrete case). The integrals of explicit functions are found using Simpson's rule; discrete case integrals may be approximated by either the trapezoidal rule or Simpson's rule.

Discrete case

Let $x_0, x_1, \ldots, x_n$ be n equally spaced points ($x_j = x_0 + jh, j = 1, 2, \ldots, n$) at which corresponding values $f(x_0), f(x_1), \ldots, f(x_n)$ of the function f(x) are known. The function itself need not be known explicitly. After input of the step size h and the values of $f(x_j)$, $j = 0, 1, \ldots, n$, then the integral

$$\int_{x_0}^{x_n} f(x)\, dx$$

may be approximated using

1. The trapezoidal rule:

$$\int_{x_0}^{x_n} f(x)\, dx \simeq \frac{h}{2} \left[f(x_0) + 2 \sum_{j=1}^{n-1} f(x_j) + f(x_n) \right]$$

2. Simpson's rule:

$$\int_{x_0}^{x_n} f(x)\, dx \simeq \frac{h}{3} \left[f(x_0) + 4f(x_1) + 2f(x_2) + \right.$$

$$\left. \ldots + 4f(x_{n-3}) + 2f(x_{n-2}) + 4f(x_{n-1}) + f(x_n) \right]$$

In order to apply Simpson's rule, n must be even. If n is not even, the calculator will halt displaying "Error" if **D** is pressed.

Explicit functions

If an explicit formula is known for the function f(x), then the function may be keyed into program memory and numerically integrated by Simpson's rule. The user must specify the endpoints a and b of the interval over which integration is to be performed, and the number of subintervals n into which the interval (a,b) is to be divided. This n must be even; if it is not, Error will be displayed. The program will go on to compute $x_0 = a$, $x_j = x_0 + jh$, $j = 1, 2, \ldots, n-1$, and $x_n = b$ where

$$h = \frac{b-a}{n}.$$

The integral $\int_a^b f(x)\,dx$ is approximated by equation (2) above, Simpson's rule.

Up to five different functions $f_i(x)$, $i = 1, \ldots, 5$, may be loaded into program memory at one time under labels 1 through 5. The function to be integrated is selected by keying in a digit 1, 2, 3, 4, or 5, and pressing **f** **E**. The function under the appropriate label will then be selected. 112 program steps are available for defining the $f_i(x)$, as well as registers R_1 through R_8, R_{S0} through R_{S9}, and the four stack registers. The functions should assume x is in the X-register upon entry. Two levels of subroutines are allowed in the functions $f_i(x)$, but recall that the only labels available are 1 through 5.

Functions $f_i(x)$ may be keyed into program memory after loading side 1 of *Numerical Integration*, or you may record these functions beforehand on a magnetic card and load them in the following manner:

1. Load side 1 of Numerical Integration.
2. Press **GTO** **.** 112.
3. Press **MERGE**.
4. Load your card with the functions $f_i(x)$.

Remarks:

Note that the function values for the discrete case $f(x_j)$, $j = 0, 1, \ldots, n$, are keyed into **B** . There are actually three routines in the program which begin with LBL B, one for $j = 0$, one for j odd, and one for j even. It is important that no other user-definable keys be pressed during the entry of the $f(x_j)$, lest the next $f(x_j)$ entered go into the wrong LBL B.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 of program.			
2	For explicit functions, go to step			
	8; for discrete case, go to step 3			
	DISCRETE			
3	Key in the spacing between			
	x-values.	h	**A**	

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
4	Repeat this step for j = 0,			
	1, ..., n: Key in the function			
	value at x_j.	$f(x_j)$	B	j
5	Compute the area by the			
	trapezoidal rule.		C	TRAP ∫
6	Compute the area by Simpson's			
	rule (n must be even).		D	SIMP ∫
7	For a new case, go to step 2.			
	EXPLICIT FUNCTIONS			
8	Load the function(s) into program			
	memory (either key them in with			
	LBL and RTN, or link from			
	step 112).			
9	Specify the function i to be			
	selected.	i(1 – 5)	f E	
10	Key in the beginning and final			
	endpoints of the integration			
	interval.	a	ENTER↑	
		b	f A	
11	Key in the number of subintervals			
	(must be even).	n	f B	
12	Compute the area by			
	Simpson's rule.		f C	$\int_a^b f_i(x)\,dx$
13	To change a, b or n, go to the			
	appropriate step; for a new case,			
	go to step 2.			

Example 1:

Given the values below for $f(x_j)$, $j = 0, 1, \ldots, 8$, compute the approximations to the integral

$$\int_0^2 f(x)\,dx$$

by the trapezoidal rule and by Simpson's rule.

The value for h is 0.25.

i	0	1	2	3	4	5	6	7	8
x_i	0	.25	.5	.75	1	1.25	1.5	1.75	2
$f(x_i)$	2	2.8	3.8	5.2	7	9.2	12.1	15.6	20

Keystrokes:	**Outputs:**
.25 [A] 2 [B] 2.8 [B] 3.8 [B]	
5.2 [B] 7 [B] 9.2 [B] 12.1 [B]	
15.6 [B] 20 [B] [C] ⟶	16.68 *** (Trapezoidal)
[D] ⟶	16.58 *** (Simpson's)

Example 2:

Find the value of

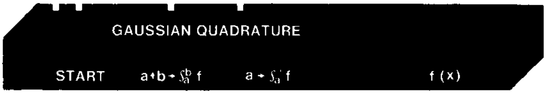
$$\int_0^{2\pi} \frac{dx}{1 - \cos x + 0.25}$$

for n = 10 and then for n = 16. Note that x is assumed to be in radians. For safety, if you work mostly in degrees, it is good programming practice to set the angular mode to radians at the beginning of the routine, then back to degrees at the end. Key the function in under LBL 1.

Keystrokes:	**Outputs:**
[GTO] [•] 112	
Switch to PRGM.	
[LBL] [1] [RAD] [COS] [1] [x⇄y] [−]	
.25 [+] [1/x] [DEG] [RTN]	
Switch to RUN.	
0 [ENTER↑] 2 [π] [×] [f] [A] 10 [f] [B]	
1 [f] [E] [f] [C] ⟶	8.22 *** (n=10)
16 [f] [B] [f] [C] ⟶	8.36 *** (n=16)

The exact solution is $\frac{8\pi}{3} = 8.38$.

GAUSSIAN QUADRATURE



This program will compute approximations for integrals over finite or infinite intervals by the six-point Gauss-Legendre quadrature method. If f(x) is the function to be integrated, then either

$$\int_a^\infty f(x)\,dx \qquad \text{or} \qquad \int_a^b f(x)\,dx$$

may be found.

The function f(x) must be explicitly known and keyed into program memory under LBL E by the user. Upon entry, the value of x will be in the X-register. 48 program steps are available for defining f(x); registers R_1 through R_9, R_{S6} through R_{S9}, R_D, R_E and the stack are also available to the user.

Equations:

$$\int_a^b f(x)\,dx \simeq \frac{b-a}{2} \sum_{i=1}^{6} w_i\, f\left(\frac{z_i(b-a)+b+a}{2}\right)$$

$$\int_a^\infty f(x)\,dx \simeq 2 \sum_{i=1}^{6} \frac{w_i}{(1+z_i)^2} f\left(\frac{2}{1+z_i} + a - 1\right)$$

where

$$\begin{aligned}
z_1 &= -z_2 = .2386191861 \\
z_3 &= -z_4 = .6612093865 \\
z_5 &= -z_6 = .9324695142 \\
w_1 &= w_2 = .4679139346 \\
w_3 &= w_4 = .360761573 \\
w_5 &= w_6 = .1713244924
\end{aligned}$$

Remarks:

If more program steps are needed to define f(x), all of LBL A (steps 001–076) may be deleted after executing it (pressing A) one time.

Reference:

Applied Numerical Methods, Carnahan, Luther and Wilks, John Wiley and Sons, 1969.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2 of program.			
2	Prepare to key in function f(x).		GTO E	
3	Switch to PRGM. Line number			
	is 177.			
4	Key in the function f(x) (need			
	not add RTN).			
5	Switch to RUN.			
6	Initialize.		A	
7	For a finite interval, key in the			
	lower and upper bounds of the			
	interval and compute the integral.	a	ENTER↑	
		b	B	$\int_a^b f(x)\,dx$
8	For an infinite interval, key in the			
	lower bound of the interval and			
	compute the integral.	a	C	$\int_a^\infty f(x)\,dx$

Example 1:

Find $\int_1^{10} \frac{dx}{x}$.

The function is $f(x) = \frac{1}{x}$; the only key required is 1/x .

Keystrokes: **Outputs:**

GTO E
Switch to PRGM.
1/x
Switch to RUN.
A 1 ENTER↑ 10 B ——→ 2.30 ***
The exact answer is ln 10.

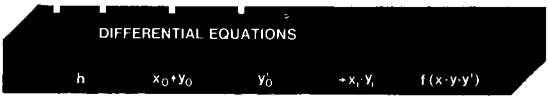
Example 2:

Find $\int_0^\infty e^{-x} x^{0.8} dx$.

Keystrokes:	**Outputs:**
GTO E	
Switch to PRGM.	
(If Example 1 has been run, delete the key 1/x .)	
CHS e^x LAST X CHS .8 y^x ×	
Switch to RUN.	
A (need not be pressed if Example 1 has been run)	
0 C ——————→	0.92 ***

The correct answer is Γ (1.8) = 0.9314.

DIFFERENTIAL EQUATIONS



This program solves first- and second-order differential equations by the fourth-order Runge-Kutta method. A first-order equation is of the form $y' = f(x,y)$, with initial values x_0, y_0; a second-order equation is of the form $y'' = f(x,y,y')$, with initial values x_0, y_0, y_0'.

In either case, the function f should be keyed into program memory under LBL E, and should assume that x and y are in the X- and Y-registers respectively; y' will be in the Z-register for second-order equations. 56 program steps are available for defining the function, as well as registers $R_1 - R_8$, $R_{S0} - R_{S9}$, and I.

The solution is a numerical solution which calculates y_i for $x_i = x_0 + ih$ ($i = 1, 2, 3, \ldots$), where h is an increment specified by the user. The value for h may be changed at any time during the program's execution. This allows solution of the equation arbitrarily close to a pole ($y \rightarrow \pm\infty$).

The values for x_i and y_i may be output in one of two ways. In its normal operation, the program will halt each time a value is calculated for x_i or y_i. The user may re-initiate execution by pressing **R/S**. Thus, in its normal use, the program outputs all results by halting and showing the result in the calculator's display. The other way to operate the program is under AUTO mode. In this case, all results are output by a PRINTx command, which means that on an HP-97, the result will appear on the printer, while on the HP-67, the program will pause briefly to display the answer. After that output, the program will automatically go on to calculate the next result.

Equations:

1st -order:

$$y_{i+1} = y_i + \frac{1}{6}(c_1 + 2c_2 + 2c_3 + c_4)$$

where

$$c_1 = hf(x_i, y_i)$$

$$c_2 = hf\left(x_i + \frac{h}{2}, y_i + \frac{c_1}{2}\right)$$

$$c_3 = hf\left(x_i + \frac{h}{2}, y_i + \frac{c_2}{2}\right)$$

$$c_4 = hf(x_i + h, y_i + c_3)$$

2^{nd} –order:

$$y_{i+1} = y_i + h\left[y_i' + \frac{1}{6}(k_1 + k_2 + k_3) \right]$$

$$y'_{i+1} = y'_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$k_1 = hf(x_i, y_i, y_i')$$

$$k_2 = hf\left(x_i + \frac{h}{2}, y_i + \frac{h}{2} y_i' + \frac{h}{8} k_1, y_i' + \frac{k_1}{2} \right)$$

$$k_3 = hf\left(x_i + \frac{h}{2}, y_i + \frac{h}{2} y_i' + \frac{h}{8} k_1, y_i' + \frac{k_2}{2} \right)$$

$$k_4 = hf\left(x_i + h, y_i + hy_i' + \frac{h}{2} k_3, y_i' + k_3 \right)$$

Remarks:

1. When inputting values for a second-order solution, the values for x_0 and y_0 must be input before the value of y_0'. All values must be input even if zero.
2. If the program is to be run for different functions, be sure that the first function is no longer in program memory when the second is keyed in. The best way to ensure this is to load the program anew before keying in each function.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2 of program.			
2	Prepare to load function f(x, y,y′)			
	under LBL E.		GTO E	
3	Switch to PRGM.			
4	Key in the function (need not			
	add RTN).			
5	Switch to RUN.			
6	Input step size.	h	A	h/2
7	Input initial values for x and y.	x_0	ENTER↑	
		y_0	B	x_0
8	For a second-order solution,			
	input initial value of y′.	y_0'	C	

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
9	For AUTO mode go to step 10;			
	for manual use, go to step 13.			
	AUTO			
10	Select AUTO mode for output			
	by Print/Pause.		f E	1.00
11	To cancel AUTO mode later.		f E	0.00
12	Output successive values of			
	x and y.		D	x_1
				y_1
				x_2
				y_2
				etc.
	Manual			
13	Output successive values of			
	x and y.		D	x_1
			R/S	y_1
			R/S	x_2
			R/S	y_2
				etc.

Example 1:

Solve numerically the first-order differential equation

$$y' = \frac{\sin x + \tan^{-1}(y/x)}{y - \ln(\sqrt{x^2+y^2})}$$

where $x_0 = y_0 = 1$. Let h = 0.5. The angular mode must be set to radians.

Keystrokes: **Outputs:**

Load side 1 and side 2 of program

GTO E

Switch to PRGM. See line 148.

RAD STO 1 x⇄y STO 2 x⇄y

→P LN STO 3 R↓ RCL 1

SIN + RCL 2 RCL 3 − ÷ DEG

Switch to RUN. Do not set Auto mode.

.5 A 1 ENTER↑ 1 B D →	1.50	(x_1)
R/S →	2.06	(y_1)
R/S →	2.00	(x_2)
R/S →	2.78	(y_2)
R/S →	2.50	(x_3)
R/S →	3.28	(y_3)

Example 2:

Solve the second-order equation

$$(1 - x^2)\, y'' + xy' = x$$

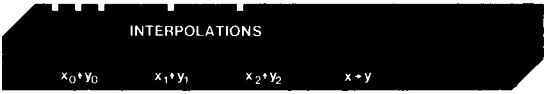
where $x_0 = y_0 = y_0' = 0$ and $h = 0.1$.

Rewrite the equation as

$$y'' = \frac{x\,(1-y')}{1-x^2} \qquad , x \neq 1$$

Keystrokes:	**Outputs:**	
Load side 1 and side 2 of program.		
GTO E		
Switch to PRGM. See line 148.		
STO 8 R↓ R↓ 1 − RCL 8 ×		
RCL 8 x^2 1 − ÷		
Switch to RUN.		
.1 A 0 ENTER↑ 0 B 0 C f E →	1.00	(AUTO mode)
DSP 4 D →	0.1000 ***	(x_1)
	0.0002 ***	(y_1)
	0.2000 ***	(x_2)
	0.0013 ***	(y_2)
	0.3000 ***	(x_3)
	0.0046 ***	(y_3)
	0.4000 ***	(x_4)
	0.0109 ***	(y_4)
	0.5000 ***	(x_5)
	0.0217 ***	(y_5)
		etc.

INTERPOLATIONS

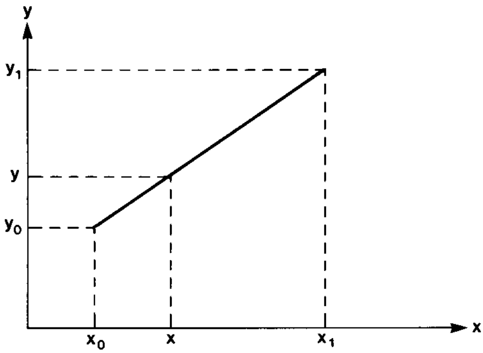


This program allows selection of one of three different interpolation routines: linear, Lagrangian, and finite difference.

Linear interpolation

If y is a function of x, let y_0 and y_1 be known function values corresponding to x_0 and x_1 respectively. Then if $x_0 < x < x_1$, the function value of x can be approximated in a linear fashion by

$$y = \frac{(x_1 - x)\, y_0 + (x - x_0)\, y_1}{x_1 - x_0}$$



Lagrangian interpolation

Given three points, (x_0, y_0), (x_1, y_1), and (x_2, y_2), the program will evaluate for an argument x the Lagrangian interpolating polynomial $P_2(x)$ of degree two which passes through the three points. Let the value of $P_2(x)$ also be denoted y.

$$P_2(x) = \sum_{i=0}^{2} L_i(x)\, y_i$$

where

$$L_i(x) = \prod_{\substack{j=0 \\ i \neq j}}^{2} \frac{(x - x_j)}{(x_i - x_j)}, \quad i=0,\ 1,\ 2$$

Finite difference interpolation

This program interpolates for data points in the region of tabulated data for uniformly spaced abscissas, with spacing h. The equation used is the backward-interpolation formula of Gauss which uses four pairs of data points and sets up the polynomial for cubic interpolation.

The equation used is:

$$y = y_3 + u\delta y_{-1/2} + \frac{1}{2}u(u+1)\,\delta^2 y_0 + \frac{1}{3!}u(u+1)(u-1)\,\delta^3 y_{-1/2}$$

The difference table is:

u	x	y			
−2	x_1	y_1			
			$y_2 - y_1$		
−1	x_2	y_2		$y_3 - 2y_2 + y_1$	
			$y_3 - y_2$		$y_4 - 3y_3 + 3y_2 - y_1$
0	x_3	y_3		$y_4 - 2y_3 + y_2$	
			$y_4 - y_3$		
1	x_4	y_4			

where

$$\delta y_{-1/2} = y_3 - y_2$$

$$\delta^2 y_0 = y_4 - 2y_3 + y_2$$

$$\delta^3 y_{-1/2} = y_4 - 3y_3 + 3y_2 - y_1$$

and

$$u = \frac{x - x_3}{h}$$

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2 of program.			
2	For linear, go to step 3; for			
	Lagrangian, go to step 7; for			
	finite difference, go to step 12.			
	LINEAR			
3	Input first point.	x_0	ENTER↑	
		y_0	A	x_0
4	Input second point.	x_1	ENTER↑	
		y_1	B	x_1

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
5	Input an x and find the inter-			
	polated y.	x	D	y
6	Repeat step 5 any number of			
	times.			
	LAGRANGIAN			
7	Input first point.	x_0	ENTER↑	
		y_0	A	x_0
8	Input second point.	x_1	ENTER↑	
		y_1	B	x_1
9	Input third point.	x_2	ENTER↑	
		y_2	C	x_2
10	Input an x and find the inter-			
	polated y, where $y = P_2(x)$.	x	D	y
11	Repeat step 10 any number of			
	times.			
	FINITE DIFFERENCE			
12	Input third abscissa.	x_3	f A	x_3
13	Input abscissa spacing.	h	f B	h
14	Input ordinates 1 through 4.	y_1	ENTER↑	
		y_2	ENTER↑	
		y_3	ENTER↑	
		y_4	f C	$\delta^2 y_0$
15	Input an x and find the inter-			
	polated y.	x	f D	y
16	Repeat step 15 any number of			
	times.			

Example 1:

The points (7.3, 1.9879) and (7.4, 2.0015) are known to lie along a curve which may be approximated by a straight line. Use linear interpolation to find approximations for the function values at 7.33 and 7.37.

Keystrokes:	Outputs:
DSP 4 7.3 ENTER↑ 1.9879 A	
7.4 ENTER↑ 2.0015 B 7.33 D ▸	1.9920 ***
7.37 D ⟶	1.9974 ***

Example 2:

The points (1, −5), (3, 1) and (10, 25) lie on a curve which is to be approximated by a second-degree polynomial. Find by Lagrangian interpolation the function values corresponding to x = 1.7 and x = 9.

Keystrokes:	Outputs:
DSP 2 1 ENTER↑ 5 CHS A	
3 ENTER↑ 1 B 10 ENTER↑ 25 C	
1.7 D ⟶	−2.94 ***
9 D ⟶	21.29 ***

Example 3:

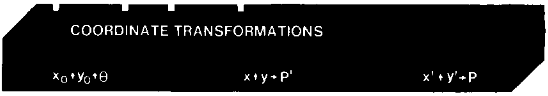
The following table lists four data points with uniformly spaced abscissas (x-values) of spacing 2.

i	1	2	3	4
x_i	−1	1	3	5
y_i	−1	2	9	30

Use finite difference interpolation to approximate the y-values for x-values of −0.5, 2.567, and 4.8.

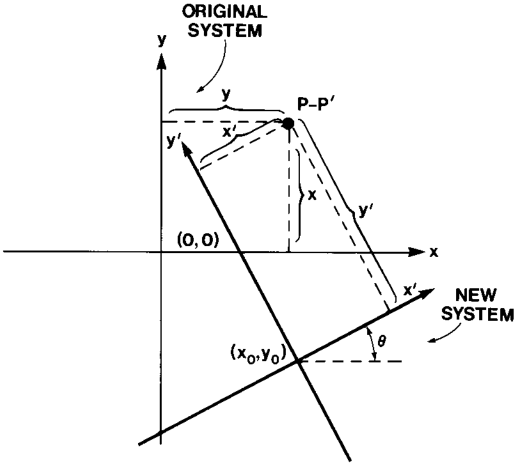
Keystrokes:	Outputs:
3 f A 2 f B 1 CHS	
ENTER↑ 2 ENTER↑ 9 ENTER↑ 30	
f C ⟶	14.00
.5 CHS f D ⟶	−0.08 ***
2.567 f D ⟶	6.64 ***
4.8 f D ⟶	26.99 ***

COORDINATE TRANSFORMATIONS



This program provides 2-dimensional and 3-dimensional coordinate translation and/or rotation.

For the 2-dimensional case, the coordinates of the origin of the translated system (x_0, y_0) and the rotation angle (θ) relative to the original system, specify the new coordinate axis. These quantities are input with the **A** key. Subsequently, points specified in the original system (x, y) may be converted to the translated rotated system (x′,y′) using the **C** key. Points in the new (x′,y′) system may be converted to points in the original (x, y) system using the **E** key.



The 3-dimensional case is analogous to the 2-dimensional case. The only important difference is the specification of the rotation. The rotation axis passes through the translated origin (x_0, y_0, z_0) and is parallel to an arbitrary direction vector ($a\vec{i}$, $b\vec{j}$, $c\vec{k}$). The sign of the rotation angle (θ) is determined by the right-hand rule and the direction of the rotation vector. For instance, the special case of 2-dimensional rotation (rotation in the (x, y) plane) could be achieved using a direction vector of (0, 0, 1) and a positive rotation angle for counter-clockwise rotations. The direction vector and angle are input using the **f** **B** key. The coordinates of the translated origin (x_0, y_0, z_0) are input using **f** **A**. Conversions from the original system (x, y, z) to the new system (x′, y′, z′) are initiated using **f** **C** while the inverse conversion is performed with **f** **E**.

Equations:

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} l_1 & m_1 & n_1 \\ l_2 & m_2 & n_2 \\ l_3 & m_3 & n_3 \end{bmatrix} \begin{bmatrix} x-x_0 \\ y-y_0 \\ z-z_0 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} l_1 & l_2 & l_3 \\ m_1 & m_2 & m_3 \\ n_1 & n_2 & n_3 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} + \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix}$$

where

$$\begin{bmatrix} l_1 & m_1 & n_1 \\ l_2 & m_2 & n_2 \\ l_3 & m_3 & n_3 \end{bmatrix} = \begin{bmatrix} a^2(1-\cos\theta) + \cos\theta & ab(1-\cos\theta) - c\sin\theta & ac(1-\cos\theta) + b\sin\theta \\ ba(1-\cos\theta) + c\sin\theta & b^2(1-\cos\theta) + \cos\theta & bc(1-\cos\theta) - a\sin\theta \\ ca(1-\cos\theta) - b\sin\theta & cb(1-\cos\theta) + a\sin\theta & c^2(1-\cos\theta) + \cos\theta \end{bmatrix}$$

Two dimensional transformations are handled as a special case of three dimensional transformation with (a, b, c) set to (0, 0, 1).

Remarks:

1. Degree mode is set when the card is loaded. However, any angular mode will work.
2. For pure translation, input zero for θ.
3. For pure rotation, input zeros for x_0, y_0, and z_0.

Reference:

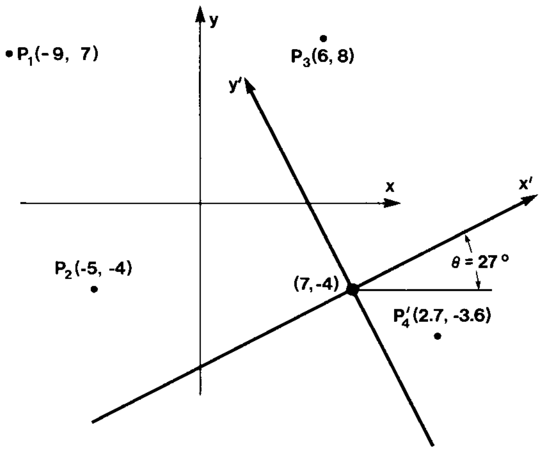
Julian, Rene S., Rotations in Three-Dimensional Space, HP-65 Users' Library Program—01368A

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2.			
2	For 2-dimensional transforma-			
	tions go to step 3.			
	For 3-dimensional transforma-			
	tions go to step 6.			
3	Input the origin of the translated			
	system and the rotation angle.	x_0	ENTER↑	
		y_0	ENTER↑	
		θ	A	1.00
4	Transform coordinates from			
	the original system to the			
	translated-rotated system.	x	ENTER↑	
		y	C	x′
				y′
	or			
	From the translated-rotated			
	system to the original system.	x′	ENTER↑	
		y′	E	x
				y
5	For a new set of coordinates, go			
	to step 4. For a new 2-dimen-			
	sional transformation, go to			
	step 3.			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
6	Input the origin of the translated			
	system.	x_0	ENTER↑	
		y_0	ENTER↑	
		z_0	f A	x_0
	and			
	Input the rotation direction			
	vector and angle.	a	ENTER↑	
		b	ENTER↑	
		c	ENTER↑	
		θ	f B	$\sqrt{a^2+b^2+c^2}$
7	Transform coordinates from			
	original system to translated			
	rotated system.	x	ENTER↑	
		y	ENTER↑	
		z	f C	x′
				y′
				z′
	or			
	From the translated-rotated			
	system to the original system.	x′	ENTER↑	
		y′	ENTER↑	
		z′	f E	x
				y
				z
8	For a new set of coordinates,			
	go to step 7.			
	For a new 3-dimensional			
	transformation go to step 6			
	(either (x_0, y_0, z_0) or (a, b, c, θ)			
	may be changed independently).			

Example 1:

The coordinate systems (x, y) and (x′, y′) are shown below:



Convert the points P_1, P_2 and P_3 to equivalent coordinates in the (x′, y′) system. Convert the point P_4' to equivalent coordinates in the (x, y) system.

Keystrokes:	**Outputs:**
7 ENTER↑ 4 CHS ENTER↑ 27 A ▶	1.00
9 CHS ENTER↑ 7 C ⟶	-9.26 *** (x'_1)
	17.06 *** (y'_1)
5 CHS ENTER↑ 4 CHS C ⟶	−10.69 *** (x'_2)
	5.45 *** (y'_2)
6 ENTER↑ 8 C ⟶	4.56 *** (x'_3)
	11.15 *** (y'_3)
2.7 ENTER↑ 3.6 CHS E ⟶	11.04 *** (x_4)
	-5.98 *** (y_4)

Example 2:

A 3-dimensional coordinate system is translated to (2.45, 4.00, 4.25). After translation, a 62.5 degree rotation occurs about the (0, −1, −1) axis. In the original system, a point had the coordinates (3.9, 2.1, 7.0). What are the coordinates of the point in the translated rotated system?

Keystrokes:	Outputs:
2.45 ENTER↑ 4.00 ENTER↑ 4.25	
f A ⟶	2.45
0 ENTER↑ 1 CHS ENTER↑ 1 CHS	
ENTER↑ 62.5 f B ⟶	1.41
3.9 ENTER↑ 2.1 ENTER↑ 7.0	
f C ⟶	3.59 *** (x′)
	0.26 *** (y′)
	0.59 *** (z′)

In the translated rotated system above, a point has the coordinate (1, 1, 1). What are the corresponding coordinates in the original system?

Keystrokes:	Outputs:
1 ENTER↑ 1 ENTER↑ 1 f E ⟶	2.91 *** (x)
	4.37 *** (y)
	5.88 *** (z)

INTERSECTIONS OF LINES AND LINES, LINES AND CIRCLE AND CIRCLES AND CIRCLES

This program calculates the point of intersection of two coplanar lines, the points of intersection of a coplanar circle and line, or the points of intersection of two coplanar circles.

Lines may be specified by two points (x, y and x′, y′), or by one point (x, y) and an angle (θ), where θ is the angle from the positive x-axis to the line. Circles are specified by their center coordinates (x_0, y_0) and the radius (r).

To find the intersection of two lines, input lines specified by two points using **f** **A** and/or **f** **B**. Input lines specified by one point and an angle using **A** and/or **B**. Calculate the point of intersection using **f** **D**. The coordinates of the point of intersection (x_p, y_p) will be output. "Error" will be displayed if you input parallel (non-intersecting) lines unless they were also vertical in which case an overflow will be generated.

Calculation of the intersections of a line and a circle requires that the line be input using **A** for point-angle representation or **f** **A** for point-point representation. The circle is input using **D**. **B**, **f** **B** and **E** must not be used during circle-line intersection calculations. Once the line parameters and circle parameters have been input, pressing **f** **D** initiates calculation of one point of intersection and **f** **E** initiates calculation of the other point. If the line is tangent to the circle, both calculated points will be identical. If the line does not intersect the circle, "Error" will be displayed.

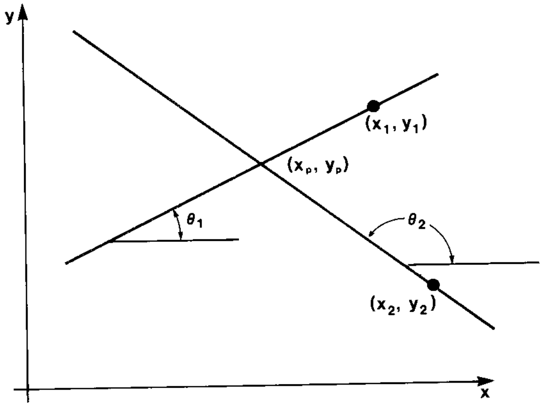
Calculation of the intersections of two circles is accomplished using the **D** and **E** keys to input the circles and **f** **D** and **f** **E** to initiate calculation of the points of intersection. If the circles are tangent both calculated points will be identical. If the two circles do not intersect "Error" will be displayed.

Equations:

Line–Line Intersection:

$$x_p = \frac{x_1 \tan \theta_1 - x_2 \tan \theta_2 + y_2 - y_1}{\tan \theta_1 - \tan \theta_2}$$

$$y_p = y_1 + (x_p - x_1) \tan \theta_1$$



Circle-Line intersections:

$$x_{p1} = x_1 + P_1\cos\theta$$

$$y_{p1} = y_1 + P_1\sin\theta$$

$$x_{p2} = x_1 + P_2\cos\theta$$

$$y_{p2} = y_1 + P_2\sin\theta$$

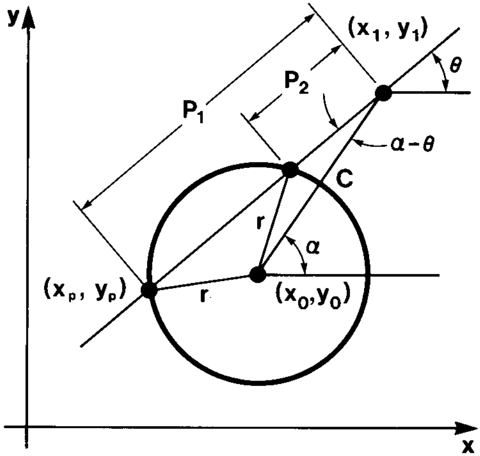
where P_1 and P_2 are the roots of

$$P^2 - 2\,D\cos(\theta-\alpha)\,P + D^2 - r^2 = 0$$

$$\theta = \tan^{-1}\left[\frac{y_2 - y_1}{x_2 - x_1}\right]$$

$$\alpha = \tan^{-1}\left[\frac{y_0 - y_1}{x_0 - x_1}\right]$$

$$D = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2}$$



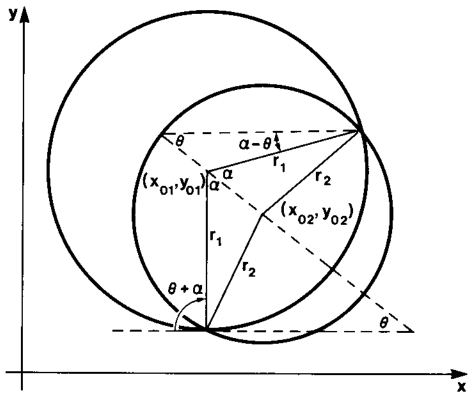
Circle-Circle intersections:

$$\begin{aligned} x_{p1} &= x_{01} + r_1 \cos(\theta + \alpha) \\ y_{p1} &= y_{01} + r_1 \sin(\theta + \alpha) \\ x_{p2} &= x_{01} + r_1 \cos(\theta - \alpha) \\ y_{p2} &= y_{01} + r_1 \sin(\theta - \alpha) \end{aligned}$$

$$\theta = \tan^{-1}\left(\frac{y_{02} - y_{01}}{x_{02} - x_{01}}\right)$$

$$\alpha = \cos^{-1}\left[\frac{D^2 + r_1^2 - r_2^2}{2Dr_1}\right]$$

$$D = \sqrt{(x_{02} - x_{01})^2 + (y_{02} - y_{01})^2}$$



Remarks:

You may specify any angular mode (degree, radian, grad) after loading the card. When the card is loaded degree mode is set.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2			
	(degree mode is set).			
2	For line/line intersections go to			
	step 3. For line/circle inter-			
	sections go to step 6. For			
	circle/circle intersections go			
	to step 10.			
	LINE/LINE			
3	Input two points on each line:			
	First point on line one.	x_1	ENTER↑	
		y_1	ENTER↑	
	Second point on line one.	x_1'	ENTER↑	
		y_1'	f A	x_1'
	First point on line two.	x_2	ENTER↑	
		y_2	ENTER↑	
	Second point on line two.	x_2'	ENTER↑	
		y_2'	f B	x_2'
	or input one point and the angle			
	of each line. Point on line one.	x_1	ENTER↑	
		y_1	ENTER↑	
	Angle of line one.	θ_1	A	x_1
	Point on line two.	x_2	ENTER↑	
		y_2	ENTER↑	
	Angle of line two.	θ_2	B	x_2
4	Calculate intersection point.		f D	x_p, y_p
5	For a new case go to step 3			
	and change either or both lines.			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
	LINE/CIRCLE			
6	Input two points on line:			
	First point on line.	x	ENTER↑	
		y	ENTER↑	
	Second point on line.	x'	ENTER↑	
		y'	f A	x'
	or input one point.	x	ENTER↑	
		y	ENTER↑	
	and angle of line.	θ	A	x
7	Input circle center	x_0	ENTER↑	
		y_0	ENTER↑	
	and radius.	r	D	x_0
8	Calculate one intersection			
	point.		f D	x_{p1}, y_{p1}
	Calculate the other intersection			
	point.		f E	x_{p2}, y_{p2}
9	For a new case go to step 6 or			
	7 and change line or circle			
	or both.			
	CIRCLE/CIRCLE			
10	Input circle one.	x_{01}	ENTER↑	
		y_{01}	ENTER↑	
		r_1	D	x_{01}
	Input circle two.	x_{02}	ENTER↑	
		y_{02}	ENTER↑	
		r_2	E	x_{02}
11	Calculate one intersection point.		f D	x_{p1}, y_{p1}
	Calculate the other intersection			
	point.		f E	x_{p2}, y_{p2}
12	For a new case go to step 10			
	and change either or both circles.			

Example 1:

Find the intersection of the vertical line specified by two points:

$$P_1 = (0, 0)$$
$$P'_1 = (0, 50)$$

And the oblique line specified by one point and an angle:

$$P_2 = (10, 20)$$
$$\theta = 45°$$

Keystrokes:	Outputs:
0 ENTER↑ 0 ENTER↑ 0 ENTER↑	
50 f A ⟶	0.00
10 ENTER↑ 20 ENTER↑ 45 B ⟶	10.00
f D ⟶	0.00 *** (x_p)
	10.00 *** (y_p)

Example 2:

Calculate the points of intersection for circles at (0, 0) radius 50 and (90, 30) radius 70.

Keystrokes:	Outputs:
0 ENTER↑ 0 ENTER↑ 50 D ⟶	0.00
90 ENTER↑ 30 ENTER↑ 70 E ⟶	90.00
f D ⟶	21.64 *** (x_{p1})
	45.07 *** (y_{p1})
f E ⟶	44.36 *** (x_{p2})
	-23.07 *** (y_{p2})

Example 3:

Find the points of intersection for a circle with center at (0, 0) and radius 50, and the line containing the points (20, 30) and (0, −10).

Keystrokes:	Outputs:
0 ENTER↑ 0 ENTER↑ 50 D ⟶	0.00

20 ENTER↑ 30 ENTER↑ 0 ENTER↑
10 CHS f A ⟶ 0.00

f D ⟶ −18.27 *** (x_{p1})
−46.54 *** (y_{p1})

f E ⟶ 26.27 *** (x_{p2})
42.54 *** (y_{p2})

CIRCLE COMPUTATIONS

This card combines two separate circle programs. One program calculates the center (x_0, y_0) and radius (r) of a circle given three non-collinear points. The other program calculates the coordinates of points on a circle (x_i, y_i), given the center and radius of the circle.

To find the center and radius of a circle, simply input three points P_1, P_2, P_3 (represented by x and y coordinates) using [f] [A], [f] [B], and [f] [C] respectively. After all three points have been input, press [f] [D] to generate the coordinates of the center (x_0, y_0) and the radius (r).

To find coordinates of points on a circle with a known center and radius, you may choose one of three options:

1. You may key in an angle θ, press [B], and calculate the coordinates of the point on the circle at angle θ, where θ is measured counterclockwise from a radius parallel to and in the direction of the positive x-axis.
2. You may manually increment around the circle one point at a time by successively pressing [E]. The outputs are angle (θ), number of the point (i), and (x, y) coordinates of the point.
3. You may automatically increment around the circle by pressing [f] [E] once. The outputs are the same as those of option two above.

For options two and three above, two input options exist:

1. An initial angle (θ) and an incremental angle ($\Delta\theta$) are specified and [C] is pressed.
2. An initial angle and the number of increments around a complete circle are specified and [D] is pressed.

Equations:

Circle determined by three points:

$$y_0 = \frac{K_2 - K_1}{N_2 - N_1}, \quad x_0 = K_2 - N_2 y_0$$

$$r = \sqrt{(x_3 - x_0)^2 + (y_3 - y_0)^2}$$

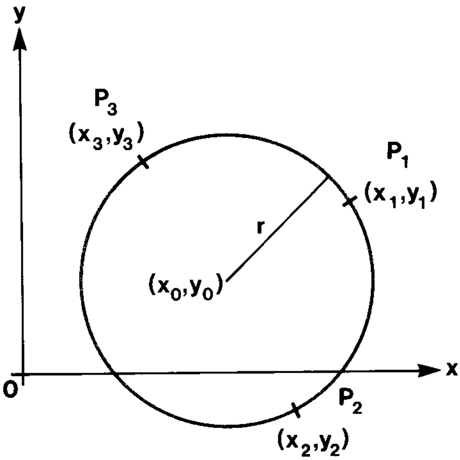
where

$$K_1 = \frac{(x_2 - x_1)(x_2 + x_1) + (y_2 - y_1)(y_2 + y_1)}{2(x_2 - x_1)}$$

$$K_2 = \frac{(x_3 - x_1)(x_3 + x_1) + (y_3 - y_1)(y_3 + y_1)}{2(x_3 - x_1)}$$

$$N_1 = \frac{y_2 - y_1}{x_2 - x_1}$$

$$N_2 = \frac{y_3 - y_1}{x_3 - x_1}$$



Points on a circle:

$$x_i = x_c + r \cos(\theta_0 + (i - 1)\,\Delta\theta)$$

$$y_i = y_c + r \sin(\theta_0 + (i - 1)\,\Delta\theta)$$

$$\Delta\theta = \frac{2\pi}{n} \quad \text{(for n evenly spaced points)}$$

$$\theta_i = \theta_0 + (i - 1)\,\Delta\theta$$

Remarks:

1. If $x_1 = x_2$ or $x_1 = x_3$ in the calculation of the center and radius of a circle, then point 1 replaces point 3, point 3 replaces point 2 and point 2 replaces point 1.

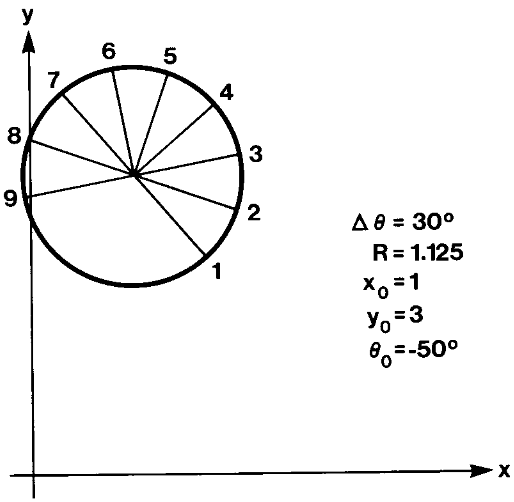
2. Degree mode is set when the card is loaded. However the program will also work for radians and grads provided the appropriate mode is set after loading the program.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2			
	(degrees mode is set).			
2	To determine a circle from three			
	points go to step 3. To generate			
	points on a circle go to step 6.			
	CIRCLE FROM THREE POINTS			
3	Input point 1.	x_1	ENTER↑	
		y_1	f A	x_1
	Input point 2.	x_2	ENTER↑	
		y_2	f B	x_2
	Input point 3.	x_3	ENTER↑	
		y_3	f C	x_3
4	Calculate center coordinates			
	and radius of circle.		f D	x_0, y_0, r
5	For a new case go to step 3 and			
	change any or all of the points.			
	POINTS ON A CIRCLE			
6	Input circle center and radius.	x_0	ENTER↑	
		y_0	ENTER↑	
		r	A	x_0
7	Optional: input an angle and			
	calculate coordinates.	θ	B	x,y
8	Input starting angle	θ_0	ENTER↑	
	and			
	angle of increment	$\Delta\theta$	C	$\Delta\theta$
	or			
	number of increments.	n	D	$\Delta\theta$

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
9	Manually increment around			
	circle by pressing E for each			
	successive increment.		E	θ_i,i,x_i,y_i
	or			
	automatically increment around			
	circle.		f E	θ_i,i,x_i,y_i
10	For a new increment size go			
	to step 8. For a new circle go to			
	step 6.			

Example 1:

Find the coordinates of the points shown on the circle below.



i	x_i	y_i
1	1.72	2.14
2	2.06	2.62
3	2.11	3.20
4	1.86	3.72
5	1.38	4.06
6	0.80	4.11
7	0.28	3.86
8	-0.06	3.38
9	-0.11	2.80

Keystrokes	Outputs
1 ENTER↑ 3 ENTER↑ 1.125 A ▸	1.00
50 CHS ENTER↑ 30 C ⟶	30.00
f E ⟶	-50.00 *** (θ)
	1.00 *** (i)
	1.72 *** (x)
	2.14 *** (y)
	-20.00 ***
	2.00 ***
	2.06 ***
	2.62 ***
	10.00 ***
	3.00 ***
	2.11 ***
	3.20 ***
	40.00 ***
	4.00 ***
	1.86 ***
	3.72 ***

70.00 ***
5.00 ***
1.38 ***
4.06 ***

100.00 ***
6.00 ***
0.80 ***
4.11 ***

130.00 ***
7.00 ***
0.28 ***
3.86 ***

160.00 ***
8.00 ***
-0.06 ***
3.38 ***

190.00 ***
9.00 ***
-0.11 ***
2.80 ***

R/S (stops program short of a complete circle.)

Example 2:

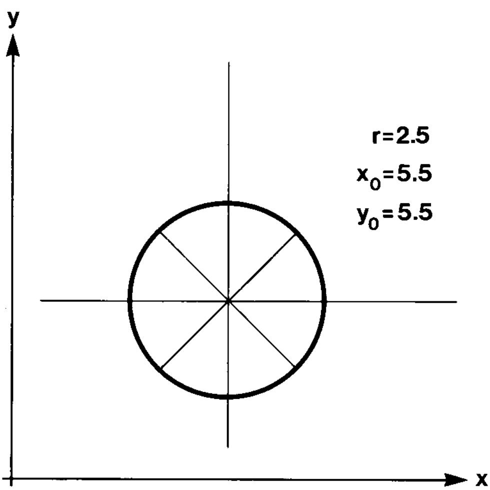
What circle contains the points (1,1), (3.5,−7.6), and (12,0.8)?

Keystrokes:	**Outputs:**
1 ENTER↑ 1 f A 3.5 ENTER↑ 7.6	
CHS f B ——→	3.50

12 ENTER↑ .8 f C ⟶	12.00
f D ⟶	6.45 *** (x_0)
	-2.08 *** (y_0)
	6.26 *** (r)

Example 3:

For the circle below calculate x and y coordinates at 4 equally spaced points, starting at 225°. Use the manual increment feature (E key). Also compute x and y at 37°.

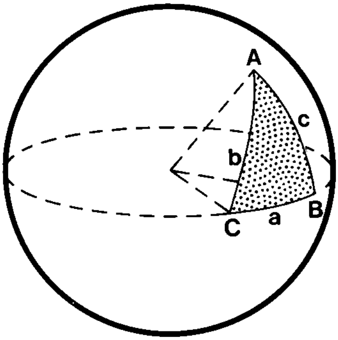


Keystrokes:	Outputs:
5.5 ENTER↑ 5.5 ENTER↑ 2.5 A ▸	5.50
225 ENTER↑ 4 D ⟶	90.00 ($\Delta\theta$)
E ⟶	225.00 ***
	1.00 ***
	3.73 ***
	3.73 ***
E ⟶	315.00 ***
	2.00 ***
	7.27 ***
	3.73 ***

E ⟶	405.00 ***
	3.00 ***
	7.27 ***
	7.27 ***
E ⟶	495.00 ***
	4.00 ***
	3.73 ***
	7.27 ***
37 B ⟶	7.50 ***
	7.00 ***

SPHERICAL TRIANGLES

This program will compute solutions to all six cases of spherical triangles, including the two ambiguous cases. In spherical triangles, as opposed to plane triangles, sides and angles have completely reciprocal qualitites. Thus a spherical triangle is well defined by the specification of its three angles. Let the angles of the triangle be A, B, C and the sides a, b, c.



Equations:

The four unambiguous cases are three sides (SSS), three angles (AAA), two sides and the included angle (SAS), and two angles and the included side (ASA). The following equations are used for the four unambiguous cases (laws of cosines):

$$\cos a = \cos b \cos c + \sin b \sin c \cos A$$
$$\cos A = - \cos B \cos C + \sin B \sin C \cos a$$

The two ambiguous cases are two sides and an opposite angle (SSA), and two angles and an opposite side (AAS). The case of SSA is equivalent to specifying a, b, A ($a \neq b$). The solution is found by the following equations:

$$\sin B = \sin b \sin A / \sin a$$

$$\tan \frac{c}{2} = \sin \left(\frac{A + B}{2} \right) \tan \left(\frac{a - b}{2} \right) / \sin \left(\frac{A - B}{2} \right)$$

$$\cos C = \frac{\cos c - \cos a \cos b}{\sin a \sin b}$$

If a < b, two solutions exist. The alternate solution is found by replacing B by its supplementary angle $\cos^{-1}(-\cos B)$. The program computes both solutions.

The case of AAS is equivalent to specifying A, B, a (A≠B). The solution is found by the following equations:

$$\sin b = \sin B \sin a / \sin A$$

$$\cot C/2 = \sin\left(\frac{a+b}{2}\right) \tan\left(\frac{A-B}{2}\right) \Big/ \sin\left(\frac{a-b}{2}\right)$$

$$\cos c = \frac{\cos C + \cos A \cos B}{\sin A \sin B}$$

If A < B, two solutions exist. The alternate solution is found by replacing b by its supplementary angle $\cos^{-1}(-\cos b)$.

In the ambiguous cases, two sets of outputs will be given if two solutions exist. Whether one or two solutions exist in these cases, the end of all output for the cases SSA and AAS is signalled by a 0.00 in the display.

For all six cases, the output is similar in format and consists of the output of every parameter of the triangle. The first value output will be the first value input, whether an angle or a side. The second output will be the adjacent value to the first output. Each successive output is adjacent to the one before, thus alternating between sides and angles. For example, if the first value input is a side, the order of the outputs will be first side, first angle, second side, second angle, third side, third angle.

Remarks:

1. AUTO mode is available to allow automatic output of all results through Print/Pause commands. If AUTO is not selected, each result will be output through a R/S.
2. The area of a spherical triangle is determined by the formula Area = $r^2 (A + B + C - \pi)$, where r is the radius of the sphere and A, B, C are in radians.
3. The program works in any angular mode. If in DEG mode, decimal degrees must be used. Note that the program sets DEG mode when read in.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2 of program.			
2	Select AUTO mode to allow			
	automatic output by			
	Print/Pause.		f E	1.00
3	To cancel AUTO mode later.		f E	0.00
4	Go to appropriate step for case			
	(SSS, SAS, SSA, AAA, ASA,			
	AAS).			
	SSS			
5	Input three sides.	S_1	ENTER↑	
		S_2	ENTER↑	
		S_3	A	OUTPUT
	SAS			
6	Input two sides and included			
	angle.	S_1	ENTER↑	
		A	ENTER↑	
		S_2	B	OUTPUT
	SSA (ambiguous)			
7	Input two sides and angle			
	opposite first side (Two sets of			
	outputs will be found if $S_1 < S_2$).	S_1	ENTER↑	
		S_2	ENTER↑	
		A	C	OUTPUT
				0.00
	AAA			
8	Input three angles.	A_1	ENTER↑	
		A_2	ENTER↑	
		A_3	D	OUTPUT

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
	ASA			
9	Input two angles and included			
	side.	A_1	ENTER↑	
		S	ENTER↑	
		A_2	E	OUTPUT
	AAS (ambiguous)			
10	Input two angles and side			
	opposite first angle (Two sets			
	of outputs will be found if			
	$A_1 < A_2$).	A_1	ENTER↑	
		A_2	ENTER↑	
		S	f A	OUTPUT
				0.00
	OUTPUT consists of the six para-			
	meters of the triangle in the order			
	First side (angle) input			
	Adjacent angle (side)			
	Adjacent side (angle)			
	Adjacent angle (side)			
	Adjacent side (angle)			
	Adjacent angle (side)			
	If two solutions exist, this schema			
	will be repeated.			

Example 1:

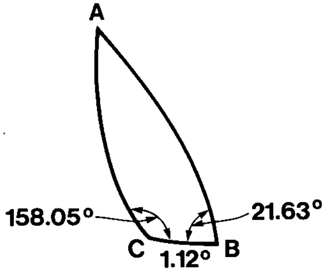
The three sides of a spherical triangle are 0.20 radians, 0.91 radians, and 0.93 radians. What are the three angles? Do not use AUTO mode.

Keystrokes:	Outputs:	
RAD .2 ENTER↑ .91 ENTER↑		
.93 A ——→	0.20	
R/S ——→	1.59	(A_1)
R/S ——→	0.91	

R/S ——→ 0.25 (A_2)
R/S ——→ 0.93
R/S ——→ 1.40 (A_3)

Example 2:

Solve the spherical triangle below for the missing parameters. Do not use AUTO mode.

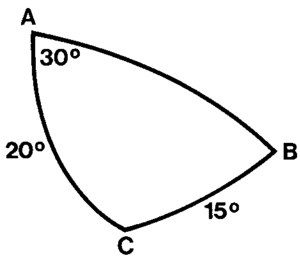


Note that this is an angle-side-angle (ASA) case.

Keystrokes:	Outputs:	
DEG 21.63 ENTER↑ 1.12 ENTER↑		
158.05 E ——→	21.63	
R/S ——→	1.12	
R/S ——→	158.05	
R/S ——→	51.90	(b)
R/S ——→	0.52	(A)
R/S ——→	52.94	(c)

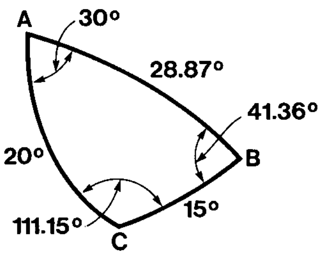
Example 3:

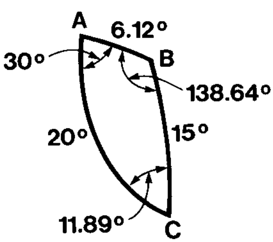
In the spherical triangle ABC below, A = 30°, a = 15°, and b = 20°. Find B, C, and c. Use AUTO mode. (Note that as this is a case of SSA, two solutions may exist.)



Keystrokes:	Outputs:	
DEG f E →	1.00	(AUTO set)
15 ENTER↑ 20 ENTER↑ 30 C →	15.00 ***	
	111.15 ***	(C)
	20.00 ***	
	30.00 ***	
	28.87 ***	(c)
	41.36 ***	(B)
	15.00 ***	
	11.89 ***	(C)
	20.00 ***	
	30.00 ***	
	6.12 ***	(c)
	138.64 ***	(B)
	0.00	(end)

The two possible solutions are pictured below.



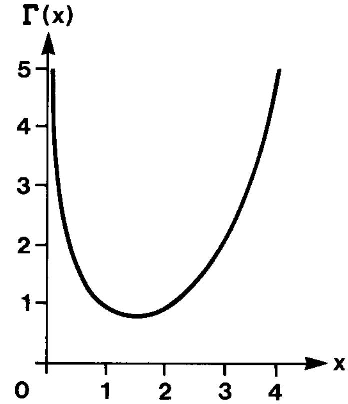


GAMMA FUNCTION

This program approximates the value of the gamma function, Γ(x), for $1 \leq x \leq 70$.

Equations:

$$\Gamma(x) = \int_0^{\infty} t^{x-1} e^{-t} dt$$



1. $\Gamma(x) = (x - 1)\,\Gamma(x - 1)$
2. For $1 \leq x \leq 2$, polynomial approximation can be used.
   $$\Gamma(x) \cong 1 + b_1 (x - 1) + b_2 (x - 1)^2 + \ldots + b_8 (x - 1)^8$$

where $b_1 = -0.577191652$, $b_2 = 0.988205891$
$b_3 = -0.897056937$, $b_4 = 0.918206857$
$b_5 = -0.756704078$, $b_6 = 0.482199394$
$b_7 = -0.193527818$, $b_8 = 0.035868343$

Remarks:

1. This program can be used to find the generalized factorial x! for $0 \leq x \leq 69$, where $x! = \Gamma(x+1)$.
2. When the value keyed in for x is an integer, Γ(x) is evaluated as the factorial of (x−1).
3. If $x < 1$, the program will halt and display "Error".

References:

Handbook of Mathematical Functions, Abramowitz and Stegun, National Bureau of Standards, 1968.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2 of program.			
2	Initialize.		A	0.00
3	Key in x and compute Γ (x).	x	B	Γ (x)
4	Repeat step 3 any number of			
	times.			

Example:

Find the gamma function for the following arguments:

5.25, 8, 3.34.

Keystrokes:	Outputs:
A 5.25 B →	35.21 ***
8 B →	5040.00 ***
3.34 B →	2.80 ***

BESSEL FUNCTIONS, ERROR FUNCTION

This card combines two separate programs in one. The first routine computes the Bessel functions $J_n(x)$ and $I_n(x)$, where n is a positive integer and $x > 0$. The second of the two routines finds the error function and complementary error function for positive arguments.

Bessel Functions

The Bessel functions $J_n(x)$ and $I_n(x)$ are computed by generating trial values T_k through the use of recurrence relations. The recurrence is begun at an index m given by

$$m = 2\ \mathrm{INT}\left[\frac{6 + \max(n, z) + \dfrac{9z}{z + 2}}{2}\right]$$

where $$z = \frac{3x}{2}$$

The initial values selected for recurrence are $T_{m+1} = 10^{-9}$, $T_{m+2} = 0$.

For the functions $J_n(x)$, each term T_k, $0 \leqslant k \leqslant m$, is computed by the relation

$$T_k(x) = \frac{2\,(k + 1)}{x} T_{k+1}(x) - T_{k+2}(x)\,,$$

beginning with $k = m$.

$J_n(x)$ is then found by dividing the term $T_n(x)$ by the normalizing constant

$$K = T_0(x) + 2 \sum_{k=1}^{m/2} T_{2k}(x)\,.$$

After calculating a $J_n(x)$, the values of $J_0(x)$ and $J_1(x)$ may also be found with very little additional computation.

For the functions $I_n(x)$, each T_k is calculated from the recurrence relation

$$T_k(x) = \frac{2(k+1)}{x} T_{k+1}(x) + T_{k+2}(x),$$

$0 \leqslant k \leqslant m$, beginning with $k = m$.

$I_n(x)$ is then found from the equation

$$I_n(x) = e^x \frac{T_n(x)}{T_0(x) + 2 \sum_{k=1}^{m} T_k(x)}$$

Error Function

The error function is defined as

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\, dt$$

and the complementary error function as

$$\operatorname{erfc}(x) = 1 - \operatorname{erf}(x).$$

For large values of x ($\geqslant 3$), the error function is very close to 1. If erfc(x) is computed as 1 - erf(x), most of the significant figures of erfc(x) will be lost for $x > 3$. Hence two different algorithms are employed in this program, one for $x \leqslant 3$ and one for $x > 3$. For $x \leqslant 3$, the error function is computed by a series sum

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} e^{-x^2} \sum_{n=0}^{\infty} \frac{2^n}{1 \cdot 3 \cdot \ldots \cdot (2n+1)} x^{2n+1}$$

and the complementary error function by

$$\operatorname{erfc}(x) = 1 - \operatorname{erf}(x).$$

For x > 3, the complementary error function is computed first, by the asymptotic expansion

$$\mathrm{erfc}(x) = \frac{1}{x\sqrt{\pi}} e^{-x^2} \left[1 + \sum_{n=1}^{\infty} \frac{(-1)^n \, 1 \cdot 3 \cdot \ldots \cdot (2n-1)}{(2x^2)^n} \right]$$

and the error function by

$$\mathrm{erf}(x) = 1 - \mathrm{erfc}(x).$$

The accuracy of the calculation of erf(x) and erfc(x) from series sums may be controlled by the user's specification of the display setting. If the display is set at DSP 6, for example, the program will halt when two successive terms of the series are equal when rounded to 6 places. Thus if the display is set to DSP N, the result will have N places of significance. Alternatively, the digit N may be keyed into the program on key **f** **E** and the display will be set automatically by the program. For $x \leq 3$, it is quite reasonable to specify DSP 9 for maximum accuracy; for $x > 3$, the series may not ever converge with DSP 9, and a safer specification would be DSP 6.

Remarks

1. The range of values $0 \leq x \leq 10^{-6}$ is out of bounds for the Bessel functions in this program. In this range, however, one may take $J_0(x) = J_0(0) = I_0(x) = I_0(0) = 1$, and $J_n(x) = J_n(0) = I_n(x) = I_n(0) = 0$, $n \neq 0$.
2. The computation of erfc(x) will halt on overflow for $x \geq 15$.

Reference

Handbook of Mathematical Functions, Abramowitz and Stegun, National Bureau of Standards, 1968.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2 of program.			
2	For Bessel functions, go to step 3;			
	for error function, go to step 6.			
	BESSEL FUNCTIONS			
3	To find J_n (x), go to step 4; to			
	find I_n (x), go to step 5.			
4	For J_n (x):			
	• Input n (n = 0, 1, 2, ...)	n	A	n
	• Input x and find J_n (x)	x	B	J_n (x)
	• (optional) Find J_0 (x) and J_1 (x)		C	J_0 (x)
			R/S	J_1 (x)
5	For I_n (x):			
	• Input n (n = 0, 1, 2, ...)	n	A	n
	• Input x and find I_n (x)	x	D	I_n (x)
	ERROR FUNCTION			
6	Specify places of accuracy			
	desired by setting display or by			
	inputting N.	N	f E	N
7	Key in x and find error function			
	and complementary error			
	function.	x	E	erf (x)
				erfc (x)

Example 1:

Find $J_5(9.2)$; also find $J_0(9.2)$ and $J_1(9.2)$. Display results to 9 places and compare to table values.

Keystrokes:	Outputs:
DSP 9 5 A 9.2 B ⟶	−0.100528623 *** $J_5(9.2)$
C ⟶	−0.136748371 $J_0(9.2)$
R/S ⟶	0.217408655 $J_1(9.2)$

The actual values from tables are $J_5(9.2) = -0.10053$, $J_0(9.2) = -0.1367483708$, and $J_1(9.2) = 0.2174086550$.

Example 2:

Find $I_3(4.7)$ and $I_3(5.0)$.

Keystrokes:	Outputs:
DSP 2 3 A 4.7 D ⟶	7.42 *** $I_3(4.7)$
5 D ⟶	10.33 *** $I_3(5.0)$

Example 3:

Find erf and erfc of 1.34 to full 9-place accuracy.

Keystrokes:	Outputs:
DSP 9 1.34 E ⟶	0.941913715 *** erf (1.34)
	0.058086285 *** erfc (1.34)

Example 4:

Find erf and erfc of 4.55 to 6 places.

Keystrokes:	Outputs:
6 f E 4.55 E ⟶	1.000000 *** erf (4.55)
	1.237404615–10 *** erfc (4.55)

HYPERBOLICS

This program computes the hyperbolic functions and their inverses. The stack is preserved during execution of any of the functions on this card. The argument, however, is not saved in the LASTx register.

Note, in the equations below, the appropriate restrictions on the values of the argument in each case.

Equations:

Hyperbolic functions

$$\sinh x = \frac{e^x - e^{-x}}{2}$$

$$\cosh x = \frac{e^x + e^{-x}}{2}$$

$$\tanh x = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

$$\operatorname{csch} x = \frac{1}{\sinh x} \quad (x \neq 0)$$

$$\operatorname{sech} x = \frac{1}{\cosh x}$$

$$\coth x = \frac{1}{\tanh x} \quad (x \neq 0)$$

Inverse Hyperbolic Functions

$$\sinh^{-1} x = \ln \left[x + (x^2 + 1)^{1/2}\right]$$

$$\cosh^{-1} x = \ln \left[x + (x^2 - 1)^{1/2}\right] \quad x \geq 1$$

$$\tanh^{-1} x = 1/2 \ln \left[\frac{1 + x}{1 - x}\right] \quad x^2 < 1$$

$$\operatorname{csch}^{-1} x = \sinh^{-1} \left[\frac{1}{x}\right] \quad x \neq 0$$

$$\text{sech}^{-1} x = \cosh^{-1}\left[\frac{1}{x}\right] \qquad 0<x\leq 1$$

$$\coth^{-1} x = \tanh^{-1}\left[\frac{1}{x}\right] \qquad x^2>1$$

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Load side 1 and side 2 of program.			
2	For hyperbolics, go to step 3;			
	for inverse hyperbolics, go to			
	step 4.			
	HYPERBOLIC FUNCTIONS			
3	Key in argument and compute			
	• hyperbolic sine	x	B	sinh x
	• hyperbolic cosine	x	C	cosh x
	• hyperbolic tangent	x	D	tanh x
	• hyperbolic cosecant	x	f B	csch x
	• hyperbolic secant	x	f C	sech x
	• hyperbolic cotangent	x	f D	coth x
	INVERSE HYPERBOLIC			
	FUNCTIONS			
4	Key in argument and compute			
	• inverse hyperbolic sine	x	A B	$\sinh^{-1}$ x
	• inverse hyperbolic cosine	x	A C	$\cosh^{-1}$ x
	• inverse hyperbolic tangent	x	A D	$\tanh^{-1}$ x
	• inverse hyperbolic cosecant	x	A	
			f B	csch^{-1} x
	• inverse hyperbolic secant	x	A	
			f C	sech^{-1} x
	• inverse hyperbolic cotangent	x	A	
			f D	$\coth^{-1}$ x

Example 1:

Evaluate the following hyperbolic functions:

sinh 2.5; cosh 3.2; tanh 1.9; csch 4.6; sech -0.25; coth -2.01.

Keystrokes:	Outputs:	
2.5 B →	6.05	(sinh 2.5)
3.2 C →	12.29	(cosh 3.2)
1.9 D →	0.96	(tanh 1.9)
4.6 f B →	0.02	(csch 4.6)
.25 CHS f C →	0.97	(sech -0.25)
2.01 CHS f D →	-1.04	(coth -2.01)

Example 2:

Evaluate the following inverse hyperbolic functions:

$\sinh^{-1}(2.4)$; $\cosh^{-1}(90)$; $\tanh^{-1}(-0.65)$; $\text{csch}^{-1}(2)$; $\text{sech}^{-1}(0.4)$; $\coth^{-1}(3.4)$.

Keystrokes:	Outputs:	
2.4 A B →	1.61	($\sinh^{-1}$ 2.4)
90 A C →	5.19	($\cosh^{-1}$ 90)
.65 CHS A D →	-0.78	($\tanh^{-1}$ -0.65)
2 A f B →	0.48	(csch^{-1} 2)
.4 A f C →	1.57	(sech^{-1} 0.4)
3.4 A f D →	0.30	($\coth^{-1}$ 3.4)

PROGRAM LISTINGS

The following listings are included for your reference. A table of keycodes and keystrokes corresponding to the symbols used in the listings can be found in Appendix E of your Owner's Handbook.

Program **Page**

Factors and Primes

Step	Key	Comment	Step	Key	Comment
001	*LBLA	Factor integer n.	057	RCLB	
002	STOB		058	-	If upper < lower, halt on Error.
003	ENT↑		059	X<0?	
004	INT		060	GTO5	
005	X≠Y?	If non-integer, halt on Error.	061	RCLA	
006	GTO5		062	INT	
007	0		063	2	Handle 2 as special case.
008	STOD	Initialize d.	064	X⇄Y	
009	X⇄Y		065	X=Y?	
010	X≤Y?	If n ≤ 0, halt on Error.	066	GTO8	
011	GTO5		067	2	
012	2		068	÷	This routine finds greatest potential prime ≤ user's input.
013	EEX		069	.	
014	9		070	5	
015	X⇄Y		071	+	
016	X>Y?		072	INT	
017	GTO5	If n > 2 x 10^9, halt on Error.	073	2	
018	CF1	F1 clear for factors.	074	x	
019	GSBa	Find factors.	075	1	
020	RTN	————————	076	-	
021	*LBLB	Lower bound for primes.	077	*LBL8	
022	STOA		078	STOE	
023	X<0?	If negative, halt on Error.	079	RCLA	Store final prime U.
024	GTO5		080	RTN	————————
025	ENT↑	This routine finds smallest potential prime ≥ user's input.	081	*LBLD	Routine to list primes.
026	INT		082	0	
027	X=Y?		083	STOD	Initialize d←0
028	GTO0		084	RCLB	
029	1		085	2	If L = 2, print 2, add 1 and go.
030	+		086	X⇄Y	
031	*LBL0		087	X=Y?	
032	2	Handle 2 as special case (only even prime).	088	GTO0	
033	X⇄Y		089	1	If L = 1, print 1, 2, add 1 and go.
034	X=Y?		090	X≠Y?	
035	GTO8		091	GTO1	
036	2		092	GSBe	If L ≠ 1 and L ≠ 2, go directly to LBL 1.
037	÷		093	+	
038	INT		094	RCLE	
039	2		095	X⇄Y	
040	x		096	X>Y?	
041	1		097	GTO4	
042	+		098	*LBL0	
043	*LBL8		099	GSBe	Output 2.
044	STOB	Store beginning prime L.	100	1	
045	STOC		101	GTO8	————
046	2	2 x 10^9 is default upper bound.	102	*LBL1	
047	EEX		103	GSBa	Begin main loop.
048	9		104	RCLB	Check for factors of current n (R_B).
049	STOE	If input ≥ 2 x 10^9, halt on Error.	105	RCLC	
050	X≤Y?		106	X=Y?	If R_B = R_C, n is prime.
051	GTO5	Flag 1 set for primes.	107	GSBe	Output n.
052	SF1		108	2	
053	RCLA		109	*LBL8	
054	RTN	————————	110	+	
055	*LBLC		111	STOC	Set n to next potential prime.
056	STOA	Upper bound for primes.	112	STOB	

REGISTERS

0	1	2	3	4	5	6	7	8	9

S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
Used	n	Potential prime	d	U	

Step	Key	Comment
113	RCLE	
114	X⇄Y	
115	X>Y?	If n > U, exit.
116	GTO4	
117	0	
118	STOD	Else loop again.
119	GTO1	------------
120	*LBLa	Subroutine called from both A & D which finds factors of n.
121	2	
122	GSB3	
123	X=0?	
124	RTN	
125	1	
126	GSB3	Check first if n divisible by 2, 3, 5, or 7.
127	X=0?	
128	RTN	
129	2	
130	GSB3	LBL 2 check for division by integers whose position in a cycle of 30 corresponds to 11, 13, 17, 19, 23, 29, 31, or 37.
131	X=0?	
132	RTN	
133	2	
134	GSB3	
135	X=0?	
136	RTN	
137	*LBL2	--------
138	4	
139	GSB3	11 (=7 +4)
140	X=0?	
141	RTN	
142	2	
143	GSB3	13 (=11 +2)
144	X=0?	
145	RTN	
146	4	
147	GSB3	17 (=13 +4)
148	X=0?	
149	RTN	
150	2	
151	GSB3	19 (=17 +2)
152	X=0?	
153	RTN	
154	4	
155	GSB3	23 (=19 +4)
156	X=0?	
157	RTN	
158	6	
159	GSB3	29 (=23 +6)
160	X=0?	
161	RTN	
162	2	31 (=29 +2)
163	GSB3	
164	X=0?	
165	RTN	
166	6	
167	GSB3	37 (=31 +6)
168	X=0?	
169	RTN	Loop again for next 30.
170	GTO2	----------
171	*LBL3	
172	RCLD	Tests if d|n.
173	+	
174	STOD	d ← d + x
175	RCLB	n
176	X⇄Y	
177	÷	n/d
178	LSTX	d, n/d
179	X>Y?	If d > n/d, then d > $\sqrt{n}$
180	GTO0	Exit.
181	X⇄Y	
182	INT	[n/d] ::= INT (n/d)
183	LSTX	n/d, [n/d]
184	X≠Y?	If non-integer, d does not divide n.
185	RTN	
186	STOB	Else n ← n/d
187	F1?	
188	CLX	If finding primes, exit.
189	F1?	
190	RTN	
191	RCLD	If factoring, output d.
192	GSBe	
193	0	Check for d a multiple factor.
194	GTO3	
195	*LBL0	-------
196	F1?	Coming here means n prime.
197	CLX	
198	F1?	If finding primes, exit.
199	RTN	
200	RCLB	Else output last n.
201	GSBe	-------
202	*LBL4	
203	CLX	Exit displaying 0.
204	F0?	
205	SPC	
206	RTN	
207	*LBLE	-----------
208	F0?	Auto toggle.
209	GTO0	
210	SF0	
211	1	
212	RTN	
213	*LBL0	
214	CF0	
215	0	
216	RTN	
217	*LBLe	-----------
218	F0?	Output routine. Print if AUTO.
219	PRTX	
220	F0?	
221	RTN	
222	R/S	
223	RTN	
224	R/S	Halt if not.

LABELS

A n → Factors	B Primes from	C Primes to	D → Primes	E AUTO?
a Factor n	b	c	d	e Output
0 Used	1 Primes loop	2 Divisor loop	3 d|n?	4 Exit
5 Non-existent	6	7	8 L = 1 or 2	9

FLAGS

0	Auto
1	Primes
2	
3	

SET STATUS

FLAGS	ON	OFF
0	☐	☒
1	☐	☒
2	☐	☒
3	☐	☒

TRIG		DISP	
DEG	☒	FIX	☒
GRAD	☐	SCI	☐
RAD	☐	ENG	☐
		n	2

GCD, LCM, Decimal to Fraction

Step	Key	Comment
001	*LBLA	GCD
002	STOB	b
003	X⇄Y	
004	STOA	a
005	1	
006	STOC	
007	CLX	
008	STOD	
009	X=Y?	If b = 0, list a as GCD.
010	GTO0	
011	STOC	
012	STOE	
013	1	s←x←0
014	STOD	
015	STOI	t←y←1
016	*LBL9	Main loop.
017	GSBe	
018	X=0?	If b = 0, list a as GCD.
019	GTO0	
020	RCLI	
021	RCLC	p←s + yq
022	STOI	y←s
023	RCL9	
024	×	
025	+	
026	STOC	s←p
027	RCLE	
028	RCLD	
029	STOE	
030	RCL9	p←t + xq
031	×	
032	+	x←t
033	STOD	t←p
034	GTO9	Loop again.
035	*LBL0	
036	RCLA	At end, a is GCD
037	X>0?	
038	GTO1	(a < 0)
039	CLX	Load stack with
040	RCLD	
041	CHS	t→Z
042	RCLC	
043	CHS	s→Y
044	RCLA	
045	CHS	
046	GTO2	GCD→X
047	*LBL1	
048	CLX	(a > 0)
049	RCLD	
050	RCLC	
051	RCLA	t
052	*LBL2	s
053	PRTX	GCD
054	R/S	Output routine.
055	R↓	
056	PRTX	(optional) Print s, t.
057	R↓	
058	PRTX	
059	SPC	
060	R↓	
061	R↓	
062	R/S	
063	*LBLB	LCM
064	STOB	b
065	X⇄Y	
066	STOA	a
067	×	
068	STOC	c = a x b
069	X=0?	If c = 0, LCM = 0
070	GTO3	
071	*LBL8	
072	GSBe	Find GCD through iteration.
073	X≠0?	
074	GTO8	At end, a = GCD.
075	RCLC	
076	RCLA	
077	÷	
078	ABS	LCM = |c ÷ GCD|
079	*LBL3	
080	PRTX	
081	SPC	
082	RTN	
083	*LBLe	Common subroutine which finds GCD if iterated until b = 0.
084	RCLA	
085	RCLA	
086	RCLB	
087	STOA	
088	÷	
089	INT	q← −INT (a/b)
090	CHS	
091	STO9	
092	RCLB	p ← a + bq = a mod b
093	×	a←b
094	+	
095	STOB	b←p
096	RTN	
097	*LBLC	Decimal to fraction.
098	0	
099	STOA	Initialize.
100	STOD	
101	R↓	
102	ENT↑	
103	STOE	R_E ←Original value.
104	1	
105	STOB	
106	STOC	
107	SF1	
108	*LBL7	
109	FIX	
110	DSP0	
111	÷	
112	LSTX	

REGISTERS

0	1	2	3	4	5	6	7	8	9
									q

S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
a; Num_{i-1}	b; Den_{i-1}	s; Num_i	t; Den_i	x; Value	y; Temp

Step	Key	Comments
113	ENT↑	
114	R↓	
115	X⇄Y	a_i
116	INT	
117	STOI	
118	x	
119	-	
120	RCLI	
121	RCLC	
122	x	
123	RCLA	
124	+	If $Num_i = 0$, clear flag 1.
125	X=0?	
126	CF1	
127	RCLC	
128	STOA	
129	R↓	
130	STOC	
131	F1?	Output Num_i.
132	GSB5	
133	CLX	
134	RCLI	
135	RCLD	
136	x	
137	RCLB	
138	+	
139	RCLD	
140	STOB	
141	R↓	
142	STOD	
143	F1?	Output Den_i.
144	GSB5	
145	RCLC	
146	X⇄Y	
147	÷	
148	DSP9	
149	F1?	Output Num_i/Den_i.
150	GSB5	
151	RCLE	Error = Calc. − orig. value.
152	-	
153	X=0?	
154	GSB6	If error = 0, halt.
155	X=0?	
156	RTN	
157	SCI	
158	F1?	Output error.
159	GSB5	
160	F1?	
161	GSB6	
162	R↓	
163	SF1	Loop again.
164	GTO7	
165	*LBLD	Recall last fraction.
166	FIX	
167	DSP0	
168	RCLC	
169	GSB5	Output Num_i.
170	RCLD	
171	GSB5	Output Den_i.
172	÷	
173	DSP9	Output Num_i/Den_i.
174	GSB5	
175	GSB6	
176	RCLE	Halt displaying error.
177	-	
178	RTN	AUTO output.
179	*LBL5	
180	F0?	If F0 set, print and rtn.
181	PRTX	
182	F0?	
183	RTN	Else halt.
184	R/S	
185	RTN	
186	*LBL6	Space if F0 set.
187	F0?	
188	SPC	
189	RTN	
190	*LBLE	AUTO toggle.
191	F0?	
192	GTO0	
193	SF0	
194	1	
195	RTN	
196	*LBL0	
197	CF0	
198	0	
199	RTN	

LABELS

A a↑b→GCD	B a↑b→LCM	C Dec→Frac	D →Last Frac	E AUTO?
a	b	c	d	e GCD/LCM
0 Exit GCD	1 GCD, a > 0	2 Output GCD	3 Exit LCM	4
5 AUTO out	6 Space	7 Dec→Frac loop	8 LCM loop	9 GCD loop

FLAGS

0	AUTO
1	$Num_i \neq 0$
2	
3	

SET STATUS

FLAGS	ON	OFF	TRIG		DISP	
0	☐	☒	DEG	☒	FIX	☒
1	☐	☒	GRAD	☐	SCI	☐
2	☐	☒	RAD	☐	ENG	☐
3	☐	☒			n	2

L03-01

Base Conversions

Step	Key	Comments
001	*LBLA	Input x_b (no. in base b to be converted).
002	STOE	
003	F0?	
004	SPC	
005	F0?	
006	PRTX	
007	1	
008	0	Default bases b & B are 10.
009	STOC	
010	STOD	
011	EEX	
012	1	
013	2	
014	STO8	
015	R↓	
016	R↓	
017	RTN	
018	*LBLB	Input base b.
019	STOD	
020	F0?	
021	PRTX	
022	F0?	
023	SPC	
024	SF2	
025	RTN	
026	*LBLC	Input base B, to which x_b is to be converted.
027	STOC	
028	F2?	
029	GTO0	
030	F0?	
031	SPC	
032	*LBL0	
033	F0?	
034	PRTX	
035	RTN	
036	*LBLD	Find x_B. Save b and B.
037	RCLC	
038	STOA	
039	RCLD	
040	STOI	
041	1	
042	0	
043	X≠Y?	Is b = 10?
044	GTO1	No, try B = 10
045	RCLC	Yes, test B > 10
046	GSBd	(Choose 10 or 100).
047	STOD	Convert
048	GSBe	Exit
049	GTO5	
050	*LBL1	b ≠ 10
051	RCLC	Is B = 10?
052	X≠Y?	No, branch.
053	GTO2	Yes, test b > 10.
054	RCLD	(Choose 10 or 100).
055	GSBd	
056	STOC	
057	GSBe	Convert
058	GTO5	Exit
059	*LBL2	
060	RCLD	Here b ≠ 10, B ≠ 10.
061	GSBd	
062	STOC	Convert x_b to x_{10}.
063	GSBe	
064	RCLE	
065	STOE	
066	RCLA	
067	STOC	
068	GSBd	Convert x_{10} to x_B.
069	STOD	
070	GSBe	
071	*LBL5	
072	PRTX	Exit
073	F0?	
074	SPC	
075	R/S	
076	*LBLd	Subroutine tests input in X-register > 10. If > 10, returns value 100. If ≤ 10, returns value 10.
077	1	
078	0	
079	STO7	
080	X⇄Y	
081	X>Y?	
082	EEX	
083	1	
084	ST×7	
085	RCL7	
086	RTN	
087	*LBLe	Main subroutine; actually converts to/from base 10.
088	0	
089	STO9	
090	STOB	
091	RCLE	
092	*LBL9	Shift right until < 1.
093	1	
094	X>Y?	
095	GTO8	
096	ST+9	R_9 keeps track of no. places shifted (exponent).
097	CLX	
098	RCLC	
099	÷	
100	STOE	
101	GTO9	
102	*LBL8	On entry, R_E contains normalized x_b: $0 < x_b < 1$.
103	RCLC	
104	RCLE	
105	×	
106	STOE	
107	EEX	
108	4	
109	+	
110	EEX	
111	4	
112	-	

REGISTERS

0	1	2	3	4	5	6	7	8	9
							10, 100	10^{12}	Used

S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
B	x_B	B, Used	b, Used	x_b, Used	b

Key	Comment
INT	
RCLB	Build up x_B.
RCLD	
x	
+	
STOB	
RCLE	
EEX	
4	
+	
EEX	
4	
-	
INT	
RCLE	
-	
ABS	
STOE	
1	
ST-9	Do not build mantissa
RCLB	beyond 10^{12}.
RCL8	
X≤Y?	
GTO4	
RCLE	
X≠0?	
GTO8	- - - - - - -
*LBL4	
RCLD	
RCL9	
Y^x	
RCLB	
x	x_B
STOB	- - - - - - - - - - - -
RTN	
*LBLE	Input new x for same b and B.
STOE	
F0?	
SPC	
F0?	
PRTX	
RCLA	Last value of B.
STOC	
RCLI	Last value of b.
STOD	
GTOD	- - - - - - - - - - - -
*LBLa	Print toggle.
F0?	
GTO0	
SF0	
1	
RTN	
*LBL0	
CF0	
0	
RTN	

LABELS					FLAGS	SET STATUS
A x_b	B b	C B	D → x_B	E x_b → x_B	0 Print	FLAGS: ON / OFF — 0 ☐ ☒; 1 ☐ ☒; 2 ☐ ☒; 3 ☐ ☒
a P?	b	c	d 10, 100	e Convert	1	TRIG: DEG ☒ GRAD ☐ RAD ☐
0 Used	1 b ≠ 10	2 b, B ≠ 10	3	4 End fE	2 Input b	DISP: FIX ☒ SCI ☐ ENG ☐ n 2
5 Used	6	7	8 Build x_B	9 Shift loop	3	

Optimal Scale for a Graph; Plotting

Step	Key	Comment
001	*LBLA	Min
002	STOE	
003	RTN	------------
004	*LBLB	
005	STOD	Max
006	RTN	------------
007	*LBLC	
008	STOC	#Tics
009	1	
010	STOI	
011	RCLD	
012	RCLE	
013	-	
014	STOB	R_B ← Max − Min
015	RCLC	
016	÷	
017	LOG	n = Floor (R_B/#Tics).
018	GSBD	
019	10^x	First guess: $\Delta = 10^n$.
020	STOA	--------
021	*LBL9	Begin loop.
022	ISZI	
023	RCLE	
024	RCLA	
025	÷	
026	GSBD	Trial bottom = Δ x Floor
027	RCLA	(Min/Δ).
028	x	
029	STO9	
030	RCLA	
031	RCLC	
032	x	
033	+	Top = Bot + Δ (#Tics).
034	STO8	
035	RCLD	
036	X≤Y?	If Max ≤ Top, exit.
037	GTO7	
038	RCLI	
039	4	Else try new Δ: If I mod 4
040	÷	= 0, Δ←1.25 Δ; else Δ←2Δ.
041	FRC	
042	X=0?	
043	GTO0	
044	2	
045	GTO6	This sets Δ to 2, 4, 5, 10, 20,
046	*LBL0	40, 50 etc., times initial
047	1	guess.
048	.	
049	2	
050	5	
051	*LBL6	
052	RCLA	
053	x	
054	STOA	Loop again.
055	GTO9	--------
056	*LBL7	Exit routine.
057	RCLB	
058	RCL8	
059	RCL9	% efficiency =
060	-	
061	÷	$\frac{\text{Max} - \text{Min}}{\text{Top} - \text{Bot}} \times 100$
062	EEX	
063	2	
064	x	
065	RCL8	
066	RCL9	Fill stack and print:
067	RCLA	Top→T
068	R↑	Bot→Z
069	PRST	Δ→Y
070	RTN	%→X
071	*LBLD	------------ Subroutine to find "floor"
072	X<0?	of x, where floor = greatest
073	GTO0	integer ≤ x.
074	INT	
075	RTN	
076	*LBL0	
077	ENT↑	
078	INT	
079	X=Y?	
080	RTN	
081	1	
082	-	
083	RTN	------------
084	*LBLa	
085	STOE	Begin x.
086	RTN	------------
087	*LBLb	
088	STOD	End x.
089	RTN	------------
090	*LBLc	
091	STOC	Step size.
092	RTN	------------
093	*LBLd	Compute (x, f_i(x)).
094	RCLE	
095	STOB	--------
096	*LBL8	
097	GSB7	Output x.
098	GSBi	
099	GSB7	Output f_i(x).
100	F0?	
101	SPC	End x.
102	RCLD	
103	RCLB	
104	RCLC	
105	+	x←x + Step.
106	X>Y?	
107	GTO0	If x > End x, exit.
108	STOB	
109	GTO8	--------
110	*LBL0	Exit.
111	CLX	
112	RTN	

REGISTERS

0	1	2	3	4	5	6	7	8	9
								Top	Bottom

S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
Δ	Max − Min; x	#Tics; Step	Max; End x	Min; Begin x	Used

Step	Key	Comments
113	*LBL7	Output subroutine
114	F0?	If AUTO set, print.
115	PRTX	
116	F0?	
117	RTN	Otherwise halt.
118	R/S	
119	RTN	
120	*LBLe	Select LBL i.
121	STOI	
122	RTN	
123	*LBLE	AUTO toggle.
124	F0?	
125	GTO0	
126	SF0	
127	1	
128	RTN	
129	*LBL0	
130	CF0	
131	0	
132	RTN	

LABELS

A Min	B Max	C #Tics	D Floor (x)	E AUTO
a Begin x	b End x	c Step	d →(x, f_i)	e i
0 Used	1	2	3	4
5	6 Used	7 Used	8 Plot loop	9 Opt. loop

FLAGS

0	AUTO
1	
2	
3	

SET STATUS

FLAGS	ON	OFF	TRIG		DISP	
0	☐	☒	DEG	☒	FIX	☒
1	☐	☒	GRAD	☐	SCI	☐
2	☐	☒	RAD	☐	ENG	☐
3	☐	☒			n	2

L05-01

Complex Operations

001	*LBLA	Input a↑b	057	STOD	
002	RCLD		058	PRTX	
003	STOB	Last b→R_B (b_1)	059	SPC	
004	R↓		060	RTN	------------
005	STOD	Present b→R_D (b_2)	061	*LBLE	Divide (÷)
006	R↓		062	RCLB	
007	RCLE	Last a→R_C (a_1)	063	RCLC	
008	STOC		064	→P	r_1 θ_1
009	R↓	Present a→R_E (a_2)	065	RCLD	
010	STOE		066	RCLE	
011	RTN	------------	067	→P	r_2 θ_2 r_1 θ_1
012	*LBLB	Add (+)	068	X⇄Y	
013	RCLC		069	CHS	
014	RCLE		070	X⇄Y	
015	+		071	1/X	$1/r_2$ $-\theta_2$ r_1 θ_1
016	STOE	$a_2 \leftarrow a_1 + a_2$	072	GTO9	------------
017	PRTX		073	*LBLa	|z|
018	RCLB		074	RCLD	
019	RCLD		075	RCLE	
020	+		076	→P	$r = \sqrt{a^2 + b^2}$
021	STOD	$b_2 \leftarrow b_1 + b_2$	077	PRTX	
022	PRTX		078	SPC	
023	SPC		079	RTN	------------
024	RTN	------------	080	*LBLb	1/z
025	*LBLC	Subtract (−)	081	RCLD	
026	RCLC		082	RCLE	
027	RCLE		083	→P	r θ
028	−		084	X⇄Y	
029	STOE	$a_2 \leftarrow a_1 - a_2$	085	CHS	
030	PRTX		086	X⇄Y	
031	RCLB		087	1/X	1/r $-\theta$
032	RCLD		088	GTO8	------------
033	−		089	*LBLc	z^n
034	STOD	$b_2 \leftarrow b_1 - b_2$	090	STOI	n→I
035	PRTX		091	RCLD	
036	SPC		092	RCLE	
037	RTN		093	→P	r θ
038	*LBLD	------------	094	RCLI	
039	RCLB	Multiply (x)	095	Y^X	
040	RCLC		096	X⇄Y	
041	→P		097	RCLI	
042	RCLD	r_1 θ_1	098	x	
043	RCLE		099	X⇄Y	
044	→P		100	GTO8	r^n $n\theta$
045	*LBL9	r_2 θ_2 r_1 θ_1	101	*LBLd	------------
046	X⇄Y		102	STOI	$z^{1/n}$
047	R↑		103	3	n→I
048	+		104	6	
049	X⇄Y		105	0	
050	R↑		106	X⇄Y	
051	x		107	÷	
052	*LBL8	$r_1 r_2$ $(\theta_1 + \theta_2)$	108	STOA	360/n→R_A
053	→R	--------	109	RCLD	
054	STOE	Output routine.	110	RCLE	
055	PRTX		111	→P	r θ
056	X⇄Y		112	RCLI	

REGISTERS

0	1	2	3	4	5	6	7	8	9

S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
360/n	b_1	a_1	b_2	a_2	n

Step	Key	Comments
113	1/X	
114	Y^x	
115	X⇄Y	
116	RCLI	
117	÷	
118	X⇄Y	$r^{1/n}$ $\frac{\theta}{n}$
119	*LBL7	
120	GSB8	Convert→R and print.
121	DSZI	
122	GTO8	Loop n times.
123	RTN	
124	*LBL8	
125	X⇄Y	
126	→P	Back→P $\left(r^{1/n}, \frac{\theta}{n}\right)$
127	X⇄Y	
128	RCLA	
129	+	$\frac{\theta}{n} + \frac{360}{n} k$
130	X⇄Y	
131	GTO7	------------
132	*LBLe	e^z
133	RAD	
134	RCLD	
135	RCLE	
136	e^x	e^a b
137	→R	
138	PRTX	
139	STOE	e^a cos b
140	X⇄Y	
141	PRTX	
142	SPC	
143	STOD	e^a sin b
144	DEG	
145	RTN	------------

LABELS					FLAGS
A a↑b	B +	C −	D x	E ÷	0
a |z|	b 1/z	c n→z^n	d n→$z^{1/n}$	e e^z	1
0 Used	1	2	3	4	2
5	6	7 Used	8 Output	9 Multiply	3

SET STATUS

FLAGS	ON	OFF	TRIG		DISP	
0	☐	☒	DEG	☒	FIX	☒
1	☐	☒	GRAD	☐	SCI	☐
2	☐	☒	RAD	☐	ENG	☐
3	☐	☒			n	2

Polynomial Solutions

Step	Key	Comment	Step	Key	Comment
001	*LBLA	5^{th} degree.	057	2	
002	4		058	ST÷6	A
003	STOE		059	ST÷7	B
004	STOI		060	RCL7	
005	GSBa	Find one real root.	061	X²	
006	RCL7		062	RCLA	
007	PRTX		063	-	
008	RCL4		064	√X	D
009	1		065	STO9	
010	GSBb	Synthetic division to find	066	X=0?	
011	GSBb	new a_i, i = 3, 2, 1, 0.	067	GTO0	- - - - - - - -
012	GSBb	- - - - - - - - - - - -	068	RCL6	
013	GSBb		069	RCL7	
014	*LBLB	4^{th} degree (quartic).	070	×	Compute C for D ≠ 0.
015	RCL2		071	RCLB	
016	STOC		072	2	
017	CHS		073	÷	
018	STO2		074	-	
019	RCL1	Compute coefficients	075	RCL9	
020	STOB	b_2, b_1, b_0 for cubic equation	076	÷	
021	RCL3	to find y_0.	077	GTO1	
022	STOD		078	*LBL0	- - - - - - - -
023	×		079	RCL6	
024	RCL0		080	X²	Compute C if D = 0.
025	STOA		081	RCLC	
026	4		082	-	
027	×		083	RCL7	
028	-		084	2	
029	STO1		085	×	
030	RCLC		086	+	
031	4		087	√X	
032	×		088	*LBL1	- - - - - - - -
033	RCLD		089	STO8	C
034	X²		090	SF0	Print roots.
035	-		091	GSB7	
036	RCLA		092	*LBL7	- - - - - - - -
037	×		093	RCL6	First time through, set
038	RCLB		094	RCL8	a_1 = A − C and a_0 = B − D.
039	X²		095	CHS	
040	-		096	STO8	
041	STO0	If b_0 = 0, Error will occur.	097	+	
042	CF0		098	STO1	
043	GSBC	Do not print roots.	099	RCL7	Second time through, set
044	F2?	Solve cubic.	100	RCL9	a_1 = A + C and a_0 = B + D.
045	GTO0	If only one real root, branch.	101	CHS	
046	RCL7		102	STO9	
047	RCL3		103	+	
048	X>Y?		104	STO0	
049	STO7	Else find largest real root	105	GSBD	
050	RCL7	from among R_3, R_4, R_7.	106	RTN	Solve quadratic $x^2 + a_1 x$
051	RCL4		107	*LBLC	$+ a_0 = 0$.
052	X>Y?		108	2	- - - - - - - - - - - -
053	STO7		109	STOE	3^{rd} degree (cubic).
054	*LBL0	$y_0 \rightarrow R_7$	110	STOI	
055	RCLD	- - - - - - - -	111	GSBa	Find one real root.
056	STO6		112	RCL7	

REGISTERS

0	1	2	3	4	5	6	7	8	9
a_0, b_0	a_1, b_1	a_2, b_2	a_3, Root	a_4, Root	$\sqrt{-D}$	A, ΔX	B, Root	C, ± 1	D
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
a_0	a_1	a_2	a_3	2 or 4	Counter

Step	Key	Comments	Step	Key	Comments
113	F0?		169	1	
114	PRTX		170	+	ΔX will be larger of the
115	RCL2		171	10^x	pair (1, 10^E).
116	1	Synthetic division to find	172	1	$R_6 \leftarrow \Delta X$
117	GSBb	new a_1, a_0.	173	X≤Y?	
118	GSBb	-------------	174	X⇄Y	-------
119	*LBLD	2nd degree (quadratic).	175	STO6	
120	RCL1		176	*LBL9	
121	CHS		177	1	
122	2	$-\frac{a_1}{2}$	178	0	
123	÷		179	ST÷6	$R_6 \leftarrow R_6/10$
124	ENT↑		180	RCL8	
125	ENT↑		181	CHS	
126	X^2	$(a_1^2/4)$	182	STO8	$R_8 \leftarrow -R_8$
127	RCL0		183	*LBL8	-------
128	-	$D = (a_1^2/4) - a_0$	184	RCL7	
129	X<0?	If $D < 0$, complex roots.	185	RCL7	
130	GTO0		186	RCL6	
131	√X	If $D \geq 0$, roots are real.	187	RCL8	
132	+		188	×	
133	STO3	Root 1 = $(-a_1/2) + \sqrt{D}$	189	+	
134	F0?		190	STO7	$R_7 \leftarrow R_7 + R_6 R_8$
135	PRTX		191	X=Y?	
136	R↑	$-a_1/2$	192	RTN	If no change, done.
137	LSTX		193	ENT↑	
138	-	$\sqrt{D}$	194	ENT↑	
139	STO4		195	GSBc	
140	F0?	Root 2 = $(-a_1/2) - \sqrt{D}$	196	X=0?	Evaluate $f(R_7)$.
141	PRTX		197	RTN	If $f(R_7) = 0$, R_7 is a root;
142	CF2	F2 clear for real roots.	198	RCL8	done.
143	RTN		199	×	
144	*LBL0	--------	200	X<0?	Else loop again.
145	CHS	$D < 0$, roots are complex.	201	GTO8	
146	√X		202	GTO9	
147	STO5	$V = \sqrt{-D}$	203	*LBLc	-----------
148	R↑		204	RCLi	Evaluate the polynomial.
149	RCL5	$u = -a_1/2$	205	+	E.g., for cubic, I = 2
150	CHS		206	×	$f(x) = ((x + a_2)x + a_1)x$
151	F0?	Stack contains −v u v u.	207	DSZI	$+ a_0$.
152	PRST		208	GTOc	
153	SF2		209	RCL0	
154	RTN	F2 set for complex roots.	210	+	
155	*LBLa	------------	211	RCLE	
156	0	Find one real root for 3rd	212	STOI	Restore I before exiting.
157	STO7	or 5th degree poly.	213	R↓	
158	RCL0		214	RTN	
159	ENT↑		215	*LBLb	-----------
160	ABS	Root will be in R_7.	216	DSZI	Synthetic division.
161	÷		217	*LBL0	E.g., for degree 5, let c_i
162	STO8	$a_0/\lvert a_0\rvert = \pm 1$	218	RCL7	be coeffs. of new poly. of
163	LSTX		219	×	degree 4: $c_3 = R_7 + a_4$;
164	RCLi	$\lvert a_0\rvert$	220	+	$c_2 = c_3 R_7 + a_3$; $c_1 = c_2$
165	ABS		221	RCLi	$R_7 + a_2$; $c_0 = c_1 R_7 + a_1$.
166	+	$\lvert a_2\rvert$ or $\lvert a_4\rvert$ (k)	222	X⇄Y	
167	LOG	Let E = $\lvert a_0\rvert$ + k	223	STOi	
168	INT		224	RTN	

LABELS

A 5th deg	B 4th deg	C 3rd deg	D 2nd deg	E
a One root	b Syn. div.	c Eval. poly.	d	e
0 Used	1 Used	2	3	4
5	6	7 Quartic	8 Loop in fA	9 Loop in fA

FLAGS

0	Print?
1	
2	Complex
3	

SET STATUS

FLAGS	ON	OFF	TRIG		DISP	
0	☒	☐	DEG	☒	FIX	☒
1	☐	☒	GRAD	☐	SCI	☐
2	☐	☒	RAD	☐	ENG	☐
3	☐	☒			n	2

4 × 4 Matrix Setup

Step	Key	Comments
001	*LBLA	Input matrix by columns:
002	5	a_{11}
003	STOI	a_{21}
004	1	a_{31}
005	STOB	a_{41},
006	STOC	a_{12}
007	*LBL9	a_{22}
008	RCLB	a_{32}
009	RCLC	a_{42},
010	1	etc.
011	0	
012	÷	
013	+	
014	R/S	
015	F0?	
016	PRTX	
017	STOi	
018	ISZI	
019	4	
020	RCLB	
021	X≠Y?	R_B is index i, R_C is index j
022	GTO0	for element a_{ij}, i, j = 1, 2, 3,
023	RCLC	4.
024	X=Y?	
025	GTOB	
026	1	
027	STOB	
028	+	
029	STOC	
030	GTO9	
031	*LBL0	
032	1	
033	+	
034	STOB	
035	GTO9	
036	*LBLB	– – – – – – – – – – – –
037	0	Begin execution.
038	STO0	Note: pivot is element of
039	1	greatest absolute value in
040	STOB	column.
041	STOD	
042	STOE	n←k←1
043	RCL5	
044	ABS	Prepare to find pivot,
045	STOC	column 1.
046	2	R_C←$\|a_{11}\|$
047	RCL6	n←2
048	GSBa	a_{21}
049	3	
050	RCL7	n←3
051	GSBa	a_{31}
052	4	
053	RCL8	n←4
054	GSBa	a_{41}
055	1	
056	RCLB	
057	X=Y?	If n = 1, no row inter-
058	GTO0	change.
059	GSBb	Else increment R_0 by n.
060	1	
061	GSBc	
062	2	Swap row n with row 1.
063	GSBc	
064	3	
065	GSBc	
066	4	
067	GSBc	
068	*LBL0	
069	RCL5	Store multipliers: m_{j1} ←
070	CHS	$-a_{j1}/a_{11}$, j = 2, 3, 4.
071	ST÷6	
072	ST÷7	
073	ST÷8	
074	9	Adjust remaining elts. by
075	STOI	multipliers: a_{ij}←a_{ij} + m_{i1}
076	GSBd	x a_{1j}, i, j = 2, 3, 4.
077	GSBd	– – – – – – – –
078	GSBd	Begin 3 x 3.
079	2	n←k←2
080	STOB	Repare to find pivot,
081	STOD	column 2.
082	P⇄S	
083	RCL0	
084	ABS	
085	STOC	R_C←$\|a_{22}\|$
086	3	n←3
087	RCL1	a_{32}
088	GSBa	
089	4	n←4
090	RCL2	a_{42}
091	GSBa	
092	P⇄S	
093	2	
094	RCLB	
095	X=Y?	If n = 2, no interchange.
096	GTO0	
097	1	
098	0	Otherwise, increment
099	x	R_0 by 10n.
100	GSBb	
101	2	
102	GSBc	
103	3	Swap row n with row 2 of
104	GSBc	3 x 3.
105	4	
106	GSBc	
107	*LBL0	
108	P⇄S	
109	RCL0	
110	CHS	Store multipliers: m_{j2} ←
111	ST÷1	$-a_{j2}/a_{22}$, j = 3, 4.
112	ST÷2	

REGISTERS

0	1	2	3	4	5	6	7	8	9
Pivots					a_{11}	a_{21}, m_{21}	a_{31}, m_{31}	a_{41}, m_{41}	a_{12}

S0	S1	S2	S3	S4	S5	S6	S7	S8	S9
a_{22}	a_{32}, m_{32}	a_{42}, m_{42}	a_{13}	a_{23}	a_{33}	a_{43}, m_{43}	a_{14}	a_{24}	a_{34}

A	B	C	D	E	I
a_{44}	i, n	j	k	±1	Used

Step	Key	Comment
113	RCL1	
114	RCL4	
115	x	Adjust remaining elts. by
116	ST+5	multipliers:
117	RCL2	$a_{ij} \leftarrow a_{ij} + m_{i2} \times a_{2j}$
118	RCL4	i, = 3, 4.
119	x	
120	ST+6	
121	RCL1	
122	RCL8	
123	x	
124	ST+9	
125	RCL2	
126	RCL8	
127	x	
128	RCLA	
129	+	
130	STOA	- - - - - - - - -
131	RCL5	Begin 2 x 2
132	ABS	If $\vert a_{43}\vert \leq \vert a_{33}\vert$, no inter-
133	RCL6	change.
134	ABS	
135	X≤Y?	
136	GTO0	Otherwise swap.
137	RCL5	
138	RCL6	
139	STO5	$a_{43} \rightleftarrows a_{33}$
140	X⇄Y	$a_{44} \rightleftarrows a_{34}$
141	STO6	
142	RCL9	
143	RCLA	
144	STO9	
145	X⇄Y	
146	STOA	
147	.	
148	4	
149	P⇄S	Increment R_0 by 0.4.
150	GSBb	
151	P⇄S	
152	*LBL0	
153	RCL5	Store multiplier:
154	CHS	$m_{43} \leftarrow -m_{43}/m_{33}$.
155	ST÷6	
156	RCL9	
157	RCL6	
158	x	
159	RCLA	
160	+	$a_{44} \leftarrow a_{44} + m_{43} \times a_{34}$
161	STOA	
162	P⇄S	DONE
163	RTN	- - - - - - - - - - - -
164	*LBLa	Used to find pivots.
165	ABS	R_c contains largest element
166	RCLC	yet found in column,
167	X>Y?	ignoring sign.
168	RTN	
169	R↓	
170	STOC	
171	R↓	
172	STOB	$n \rightarrow R_B$
173	RTN	- - - - - - - - - - - -
174	*LBLb	Store pivot info in R_0.
175	ST+0	CHS of R_E for each swap.
176	RCLE	- - - - - - - - - - - -
177	CHS	Swap j^{th} (R_c) elts. in rows
178	STOE	m (R_B) and k (R_D).
179	RTN	
180	*LBLc	
181	STOC	
182	RCLD	
183	RCLC	
184	GSBE	a_{kj}
185	RCLB	
186	RCLC	a_{mj} (→t)
187	GSBE	
188	X⇄Y	$a_{mj} \leftarrow a_{kj}$
189	STOi	
190	X⇄Y	
191	RCLD	
192	RCLC	
193	4	
194	x	
195	+	
196	STOI	
197	R↓	
198	STOi	
199	RTN	
200	*LBLE	$a_{kj} \leftarrow t$
201	4	- - - - - - - - - - - -
202	x	Given i in Y, j in X,
203	+	recalls a_{ij}.
204	STOI	
205	CLX	
206	RCLi	
207	RTN	
208	*LBLd	
209	RCLi	- - - - - - - - - - - -
210	STOB	Used to adjust each
211	ISZI	column of 3 x 3 sub-
212	RCL6	matrix by multipliers in
213	GSBe	R_6, R_7, and R_8.
214	RCL7	
215	GSBe	
216	RCL8	
217	GSBe	
218	RTN	
219	*LBLe	
220	RCLB	
221	x	- - - - - - - - - - - -
222	ST+i	R_B contains a_{1j}, j = 2, 3, 4.
223	ISZI	
224	RTN	Goes down column.

LABELS

A Input	B Execute	C	D	E RCL a_{ij}
a Find pivot	b Store pivot	c $a_{mj} \rightleftarrows a_{kj}$	d Adjust 3x3	e Col. Mult.
0 Used	1	2	3	4
5	6	7	8	9 Input loop

FLAGS

0	Print?
1	
2	
3	

SET STATUS

FLAGS	ON	OFF	TRIG		DISP	
0	☐	☒	DEG	☒	FIX	☒
1	☐	☒	GRAD	☐	SCI	☐
2	☐	☒	RAD	☐	ENG	☐
3	☐	☒			n	1

4 × 4 Matrix Solutions

Step	Key	Comments
001	*LBLA	Find determinant.
002	RCLE	
003	RCL5	$R_E = \pm 1$ depending on
004	×	number of row interchanges.
005	P⇄S	
006	RCL0	
007	×	
008	RCL5	
009	×	$\lvert Det \rvert = a_{11}\, a_{22}\, a_{33}\, a_{44}$
010	P⇄S	
011	RCLA	
012	×	
013	RTN	- - - - - - - - - - - -
014	*LBLB	Input vector
015	ST04	$\underset{\sim}{b} = (b_1, b_2, b_3, b_4)$
016	R↓	
017	ST03	
018	R↓	
019	ST02	
020	R↓	
021	ST01	
022	*LBLa	- - - - - - - - - - - -
023	RCL0	Solves $A\underset{\sim}{x} = \underset{\sim}{b}$
024	1	First find $L\underset{\sim}{b}$.
025	0	
026	STOD	Pick off units digit of R_0
027	÷	(= n).
028	FRC	
029	RCLD	
030	×	
031	INT	
032	X=0?	If n = 0, no interchange.
033	GTO0	
034	STOI	If n ≠ 0, swap b_n and b_1.
035	RCLi	
036	RCL1	
037	STOi	
038	X⇄Y	
039	STO1	
040	*LBL0	
041	RCL1	
042	RCL6	
043	×	
044	ST+2	$b_2 \leftarrow b_2 + m_{21}\, b_1$
045	RCL1	
046	RCL7	
047	×	
048	ST+3	$b_3 \leftarrow b_3 + m_{31}\, b_1$
049	RCL1	
050	RCL8	
051	×	
052	ST+4	$b_4 \leftarrow b_4 + m_{41}\, b_1$
053	RCL0	- - - - - - - -
054	RCLD	Begin k = 2.
055	÷	Pick off tens digit of R_0
056	INT	(= n).
057	X=0?	If n = 0, no interchange.
058	GTO0	
059	STOI	If n ≠ 0, swap b_n and b_2.
060	RCLi	
061	RCL2	
062	STOi	
063	X⇄Y	
064	ST02	
065	*LBL0	
066	P⇄S	
067	RCL2	
068	RCL1	
069	P⇄S	
070	RCL2	$b_3 \leftarrow b_3 + m_{32}\, b_2$
071	×	
072	ST+3	
073	CLX	
074	RCL2	
075	×	
076	ST+4	$b_4 \leftarrow b_4 + m_{42}\, b_2$
077	RCL0	- - - - - - - -
078	FRC	Begin k = 3.
079	RCLD	Pick off fractional digit
080	×	of R_0 (= n).
081	X=0?	
082	GTO0	If n = 0, no interchange.
083	STOI	
084	RCLi	If n ≠ 0, swap b_n and b_3.
085	RCL3	
086	STOi	
087	X⇄Y	(n can only be 4)
088	STO3	
089	*LBL0	
090	P⇄S	
091	RCL6	
092	P⇄S	
093	RCL3	
094	×	
095	ST+4	$b_4 \leftarrow b_4 + m_{43}\, b_3$
096	RCLA	- - - - - - - -
097	ST÷4	Now solve $U\underset{\sim}{x} = L\underset{\sim}{b}$
098	RCL4	$b_4 \leftarrow b_4 / a_{44}$
099	CHS	$t \leftarrow -b_4$
100	STOB	
101	P⇄S	
102	RCL5	
103	STOC	
104	RCL9	$R_C \leftarrow a_{33}$
105	RCL8	a_{34}
106	RCL7	a_{24}
107	P⇄S	a_{14}
108	RCLB	
109	×	
110	ST+1	$b_1 \leftarrow b_1 - a_{14}\, b_4$
111	CLX	
112	RCLB	

REGISTERS

0 Pivots	1 b_1	2 b_2	3 b_3	4 b_4	5 a_{11}	6 m_{21}	7 m_{31}	8 m_{41}	9 a_{12}
S0 a_{22}	S1 m_{32}	S2 m_{42}	S3 a_{13}	S4 a_{23}	S5 a_{33}	S6 m_{43}	S7 a_{14}	S8 a_{24}	S9 a_{34}
A a_{44}	B Used	C Used	D 10	E ±1	I Used				

Step	Key	Comments
113	x	$b_2 \leftarrow b_2 - a_{24}\, b_4$
114	ST+2	
115	CLX	$b_3 \leftarrow b_3 - a_{34}\, b_4$
116	RCLB	- - - - - - - -
117	x	$b_3 \leftarrow b_3 - b_3/a_{33}$
118	ST+3	
119	RCLC	$R_B \leftarrow -b_3$
120	ST÷3	
121	RCL3	
122	CHS	
123	STOB	
124	P⇄S	
125	RCL8	$R_c \leftarrow a_{22}$
126	STOC	a_{23}
127	RCL4	a_{13}
128	RCL3	
129	P⇄S	
130	RCLB	
131	x	
132	ST+1	$b_1 \leftarrow b_1 - a_{13}\, b_3$
133	CLX	
134	RCLB	
135	x	
136	ST+2	$b_2 \leftarrow b_2 - a_{23}\, b_3$
137	RCLC	- - - - - - - -
138	ST÷2	$b_2 \leftarrow b_2/a_{22}$
139	RCL9	
140	RCL2	
141	CHS	
142	x	
143	ST+1	$b_1 \leftarrow b_1 - a_{12}\, b_2$
144	RCL5	- - - - - - - -
145	ST÷1	$b_1 \leftarrow b_1/a_{11}$
146	F0?	
147	SPC	
148	RCL1	Output
149	GSB5	b_1
150	RCL2	b_2
151	GSB5	b_3
152	RCL3	b_4
153	GSB5	- - - - - - - - - - - -
154	RCL4	Output routine.
155	*LBL5	Print for AUTO.
156	F0?	
157	PRTX	
158	F0?	
159	RTN	
160	R/S	Halt otherwise.
161	RTN	- - - - - - - - - - - -
162	*LBLC	Compute inverse by 4 calls
163	GSBc	to LBL a, each one solving
164	1	$A\underset{\sim}{x} = \underset{\sim}{I_j}$, j = 1, 2, 3, 4,
165	STO1	
166	GSBa	
167	GSBc	
168	1	
169	STO2	
170	GSBa	where $\underset{\sim}{I_j}$ is the column
171	GSBc	vector of length 4 with
172	1	$b_i = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$
173	STO3	
174	GSBa	
175	GSBc	i = 1, 2, 3, 4
176	1	
177	STO4	
178	GSBa	
179	CLX	
180	RTN	- - - - - - - - - - - -
181	*LBLc	
182	CLX	Set all $b_i = 0$.
183	STO1	
184	STO2	
185	STO3	
186	STO4	
187	RTN	- - - - - - - - - - - -
188	*LBLE	
189	F0?	AUTO toggle.
190	GTO0	
191	SF0	
192	1	
193	RTN	
194	*LBL0	
195	CF0	
196	0	
197	RTN	- - - - - - - - - - - -

LABELS

A →Det	B Input $\underset{\sim}{b}$	C Inverse	D	E AUTO?
a Solve	b	c Clear $\underset{\sim}{b}$	d	e
0 Used	1	2	3	4
5 Output	6	7	8	9

FLAGS

0	AUTO
1	
2	
3	

SET STATUS

FLAGS	ON	OFF
0	☐	☒
1	☐	☒
2	☐	☒
3	☐	☒

TRIG	
DEG	☒
GRAD	☐
RAD	☐

DISP	
FIX	☒
SCI	☐
ENG	☐
n	2

L08-01

Solution to f(x) = 0 on an Interval

Instruction	Comment
*LBLA	b
STOB	
GSBE	f(b)
STO8	
RTN	------------
*LBLB	
STOA	c
STOC	
GSBE	
STO8	f(c)
STO9	------------
RTN	
*LBLC	TOL
STOE	
RTN	------------
*LBLD	
RCL8	If f(b) = 0, exit.
X=0?	
GTO5	
RCLB	
RCLC	
-	
ABS	
RCLE	If TOL > |b − c|, exit.
X>Y?	
GTO5	
2	
÷	
EEX	
CHS	
9	TOL1 = 10^{-9}b + ½ TOL.
RCLB	
×	
+	
STOI	
RCL8	
RCL0	Linear interpolation (secant
RCL8	method):
-	
RCLA	$b' = b - \frac{f(b)}{\frac{f(a) - f(b)}{a - b}}$
RCLB	
-	
÷	
÷	b′ may be next b.
RCLB	
X⇄Y	
-	
STOD	Test if b′ ∈ [b, c].
RCLB	
-	
RCLC	$s = \frac{b' - b}{c - b}$
RCLB	
-	
÷	
X<0?	If s < 0, reject b′.
GTO1	
1	If s ≥ 1, reject b′.
X≤Y?	
GTO1	
RCLB	
RCLC	
-	
ABS	
4	
÷	If b′ is closer than
RCLD	
RCLC	$\frac{\|b - c\|}{4}$ to c, then reject b′.
-	
ABS	
X≤Y?	
GTO1	
RCLI	
RCLD	
RCLB	
-	If |b′ − b| ≤ TOL1, b′←b+
ABS	TOL1 × sgn (c − b).
X>Y?	
GTO2	
X⇄Y	
RCLC	
RCLB	
-	
ENT↑	
ABS	
÷	
×	
RCLB	
+	
STOD	
GTO2	
*LBL1	--------
RCLB	Reject b′, set
RCLC	
+	$b' = \frac{b + c}{2}$, i.e., midpoint
2	
÷	of [b, c].
STOD	--------
*LBL2	Set new values for next iteration.
RCLB	
STOA	a←b
RCL8	
STO0	f(a)←f(b)
RCLD	
STOB	b←b′
GSBE	
STO8	f(b)←f(b′)
RCL9	
×	
X<0?	If f(b) × f(c) < 0, leave c unchanged.
GTO3	
RCLA	Else replace c←a.
STOC	

REGISTERS

0	1	2	3	4	5	6	7	8	9
f(a)								f(b)	f(c)
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
a	b	c	b′	TOL	TOL1

Step	Instruction	Comment
113	RCL8	f(c)←f(a)
114	ST09	---------
115	*LBL3	
116	RCL9	
117	ABS	
118	RCL8	
119	ABS	If |f(b)| < |f(c)|, loop again.
120	X≤Y?	
121	GTOD	
122	RCLB	Else swap b and c and set
123	RCLC	a←(new) c.
124	STOB	
125	X⇄Y	
126	STOC	
127	STOA	
128	RCL8	
129	RCL9	
130	ST08	
131	X⇄Y	
132	ST09	
133	ST08	
134	GTOD	Loop again.
135	*LBL5	---------
136	RCLB	Display root.
137	RTN	-------------
138	*LBLE	
139	RTN	User-defined f(x).

LABELS

A b	B c	C TOL	D → root	E f(x)
a	b	c	d	e
0	1 Reject b′	2 New a, b, c	3 End loop	4
5 Exit	6	7	8	9

FLAGS

0	
1	
2	
3	

SET STATUS

FLAGS	ON	OFF	TRIG		DISP	
0	☐	☒	DEG	☒	FIX	☒
1	☐	☒	GRAD	☐	SCI	☐
2	☐	☒	RAD	☐	ENG	☐
3	☐	☒			n	2

Numerical Integration

Step	Key	Comment	Step	Key	Comment
001	*LBLA	Input h.	057	*LBL7	
002	STOD		058	2	If n is not even, display
003	RTN	------------	059	÷	"Error" by call to E.
004	*LBLB	Input $f(x_0)$	060	FRC	
005	STO0	R_0 contains TRAP Σ.	061	X≠0?	
006	STO9	R_9 contains SIMP Σ.	062	GTOE	------------
007	0	n = 0	063	RTN	Compute Simpson area.
008	STOC		064	*LBLc	
009	*LBL9		065	RCLA	
010	RTN	------------	066	GSBi	$R_0 \leftarrow f(a)$
011	*LBLB	Input $f(x_j)$, j odd.	067	STO0	
012	STOA	$R_0 \leftarrow R_0 + 2\,f(x_j)$	068	RCLB	
013	GSB6		069	GSBi	
014	ENT↑		070	ST+0	$R_0 \leftarrow R_0 + f(b)$
015	+		071	RCLB	
016	ST+9	$R_9 \leftarrow R_9 + 4\,f(x_j)$	072	RCLA	
017	RCLC		073	STOE	Set initial x to a.
018	1		074	-	
019	+		075	RCLC	
020	STOC	$n \leftarrow n + 1$	076	÷	
021	RTN		077	STOD	$h \leftarrow (b - a)/n$
022	*LBLE	------------	078	0	
023	STOA	Input $f(x_j)$, j even.	079	STO9	R_9 will count to n.
024	GSB6	$R_0 \leftarrow R_0 + 2\,f(x_j)$	080	*LBL8	--------
025	ST+9		081	RCLD	
026	RCLC		082	RCLE	
027	1	$R_9 \leftarrow R_9 + 2\,f(x_j)$	083	+	
028	+		084	STOE	$x \leftarrow x + h$
029	STOC	$n \leftarrow n + 1$	085	GSBi	
030	GTO9	Exit	086	GSB6	
031	*LBLC	------------	087	ST+0	
032	2	Compute trapezoidal area.	088	2	$R_0 \leftarrow R_0 + 4\,f(x)$
033	RCL0		089	ST+9	
034	GTO0		090	RCLC	
035	*LBLD	------------	091	RCL9	
036	RCLC	Compute Simpson area.	092	X=Y?	If R_9 = n, exit.
037	GSB7		093	GTO0	
038	3	Test n even.	094	RCLD	
039	RCL9		095	RCLE	
040	*LBL0		096	+	
041	RCLA		097	STOE	
042	-	2 $f(x_n)$ were added, so sub-	098	GSBi	$x \leftarrow x + h$
043	*LBLd	tract $f(x_n)$.	099	GSB6	
044	RCLD		100	GTO8	$R_0 \leftarrow R_0 + 2\,f(x)$
045	×		101	*LBL0	--------
046	X⇄Y		102	3	
047	÷		103	RCL0	Area = $\frac{h}{3} \times R_0$
048	PRTX		104	GTOd	
049	RTN	Area	105	*LBL6	
050	*LBLa	------------	106	ENT↑	------------
051	STOB	Input a and b .	107	+	Subroutine.
052	X⇄Y	Store b.	108	ST+0	
053	STOA		109	RTN	
054	RTN	Store a.	110	*LBLe	$R_0 \leftarrow R_0 + 2\,f(x)$
055	*LBLb	------------ Input n.	111	STOI	------------ Input i to select function
056	STOC		112	RTN	1–5.

REGISTERS

0	1	2	3	4	5	6	7	8	9
Used									Used
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
$f(x_j)$, a	b	n	h	x	Function i

LABELS					FLAGS
A h	B $f(x_j)$	C →TRAP ∫	D →SIMP ∫	E None	0
a a↑b	b n	c →∫ f_i	d Output	e i	1
0 Used	1	2	3	4	2
5	6 2 f(x)	7 Test n	8 Loop fc	9 Loop B	3

SET STATUS

FLAGS	ON	OFF
0	☐	☒
1	☐	☒
2	☐	☒
3	☐	☒

TRIG	
DEG	☒
GRAD	☐
RAD	☐

DISP	
FIX	☒
SCI	☐
ENG	☐
n	2

Gaussian Quadrature

Step	Key	Comment
001	*LBLA	
002	P⇄S	
003	.	
004	2	
005	3	
006	8	
007	6	$z_1 = -z_2$
008	1	
009	9	
010	1	
011	8	
012	6	
013	1	
014	STO0	
015	.	
016	4	
017	6	
018	7	
019	9	
020	1	$w_1 = w_2$
021	3	
022	9	
023	3	
024	4	
025	6	
026	STO1	
027	.	
028	6	
029	6	
030	1	
031	2	$z_3 = -z_4$
032	0	
033	9	
034	3	
035	8	
036	6	
037	5	
038	STO2	
039	.	
040	3	
041	6	
042	0	
043	7	
044	6	$w_3 = w_4$
045	1	
046	5	
047	7	
048	3	
049	STO3	
050	.	
051	9	
052	3	
053	2	
054	4	$z_5 = -z_6$
055	6	
056	9	
057	5	
058	1	
059	4	
060	2	
061	STO4	
062	.	
063	1	
064	7	
065	1	
066	3	$w_5 = w_6$
067	2	
068	4	
069	4	
070	9	
071	2	
072	4	
073	STO5	
074	P⇄S	
075	CLX	
076	RTN	
077	*LBLB	Integral from a to b.
078	STOB	b
079	X⇄Y	
080	STOC	a
081	-	
082	2	
083	÷	
084	STOA	$R_A \leftarrow (b-a)/2$
085	RCLC	
086	RCLB	
087	+	
088	2	
089	÷	
090	STOB	$R_B \leftarrow (b+a)/2$
091	0	
092	STO0	R_0 will accumulate Σ.
093	1	
094	0	Set I to point at z_1.
095	STOI	
096	GSBb	Each call to b computes 2 terms for Σ.
097	GSBb	
098	GSBb	
099	RCL0	
100	RCLA	
101	×	$\frac{b-a}{2} \Sigma w_i f(x_i)$
102	PRTX	
103	RTN	
104	*LBLb	
105	RCLi	Computes terms of Σ.
106	ISZI	z_j (j = 1, 3, 5)
107	STOC	I points to w_j
108	CHS	$R_c \leftarrow z_j$
109	RCLA	$z_{j+1} = -z_j$
110	×	Compute term j + 1
111	RCLB	$x_{j+1} \leftarrow [z_{j+1}(b-a)+b+a]/2$.
112	+	

REGISTERS

0	1	2	3	4	5	6	7	8	9
Σ									

S0	S1	S2	S3	S4	S5	S6	S7	S8	S9
z_1	w_1	z_3	w_3	z_5	w_5				

A	B	C	D	E	I
(b − a)/2	(b + a)/2	Used			10−15

Key	Comment
GSBE	$w_j = w_{j+1}$
RCLi	$w_{j+1}\ f(x_{j+1})$
×	
ST+0	
RCLC	
RCLA	Compute term j.
×	
RCLB	
+	
GSBE	$w_j\ f(x_j)$
RCLi	
×	
ST+0	I points at z_{j+2}.
ISZI	
RTN	
*LBLC	Integral from a to ∞.
STOA	a
0	
STO0	R_0 will accumulate Σ.
1	
0	I points at z_1.
STOI	
GSBc	Each call to c computes 2
GSBc	terms for Σ.
GSBc	
RCL0	
2	
×	
PRTX	$2\Sigma \frac{w_i}{(1+z_i)^2} f(x_i)$
RTN	
*LBLc	
RCLi	Computes terms of Σ.
ISZI	z_i (j = 1, 3, 5).
STOC	I points to w_j.
CHS	$R_c \leftarrow z_j$
GSB9	$z_{j+1} = -z_j$
GSBE	$f(x_{j+1})$
GSB8	
RCLC	
GSB9	z_j
GSBE	$f(x_j)$
GSB8	
ISZI	
RTN	I points to z_{j+2}.
*LBL9	
1	Computes argument.
+	
STOB	$x_j = \frac{2}{1+z_j} + a - 1$
2	
X⇄Y	
÷	
RCLA	
+	
1	
−	
RTN	
*LBL8	Finds one term.
RCLi	
×	$\frac{w_j}{(1+z_j)^2} f(x_j)$
RCLB	
X^2	
÷	
ST+0	
RTN	48 steps reserved for user-
*LBLE	defined function f(x).
RTN	

Also available are $R_1 - R_9$, R_D, R_E, and stack.

LABELS					FLAGS
A START	B $a \uparrow b \rightarrow \int_a^b f$	C $a \rightarrow \int_a^\infty f$	D	E User f(x)	0
a	b Σ terms (B)	c Σ terms (C)	d	e	1
0	1	2	3	4	2
5	6	7	8 Used in c	9 Used in c	3

SET STATUS		
FLAGS	TRIG	DISP
ON OFF		
0 ☐ ☒	DEG ☒	FIX ☒
1 ☐ ☒	GRAD ☐	SCI ☐
2 ☐ ☒	RAD ☐	ENG ☐
3 ☐ ☒		n 2

Differential Equations

Step	Key	Comment
001	*LBLA	h/2
002	2	
003	÷	------------
004	STOA	
005	RTN	F1 clear for 1st-order.
006	*LBLB	y_0
007	CF1	
008	STOC	x_0
009	R↓	
010	STOB	------------
011	RTN	F1 set for 2nd -order.
012	*LBLC	y_0'
013	SF1	
014	STOD	------------
015	RTN	Compute solution.
016	*LBLD	Branch for 2nd-order.
017	F1?	--------
018	GTOd	1st-order solution.
019	*LBL9	
020	RCLC	
021	RCLB	
022	GSBE	$k_1/2$
023	STOE	
024	RCLC	$y_i + (k_1/2)$
025	+	
026	RCLB	
027	RCLA	$x_i + (h/2)$
028	+	
029	GSBE	$k_2/2$
030	STO0	
031	RCLC	$y_i + (k_2/2)$
032	+	
033	RCLB	
034	RCLA	$x_i + (h/2)$
035	+	
036	GSBE	$k_3/2$
037	ST+0	
038	ENT↑	
039	+	
040	RCLC	$y_i + k_3$
041	+	
042	RCLB	
043	RCLA	
044	ENT↑	h
045	+	
046	STOD	$x_i + h$
047	+	$k_4/2$
048	GSBE	
049	RCLE	
050	+	
051	RCL0	
052	ENT↑	
053	+	
054	+	
055	3	$\Delta = \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$
056	÷	

Step	Key	Comment
057	RCLC	
058	+	
059	STOC	$y_{i+1} = y_i + \Delta$
060	RCLB	
061	RCLD	$x_{i+1} = x_i + h$
062	+	
063	STOB	
064	GSB8	
065	RCLC	
066	GSB8	
067	F0?	Loop again.
068	SPC	--------
069	GTO9	2nd-order solution.
070	*LBLd	y_0'
071	RCLD	y_0
072	RCLC	x_0
073	RCLB	
074	GSBE	$k_1/2$
075	STOE	
076	STO9	
077	RCLA	
078	GSBa	$k_2/2$
079	STO0	
080	STO9	
081	RCLA	
082	RCL9	
083	RCLD	
084	+	
085	RCLE	$k_3/2$
086	GSBb	
087	ST+0	
088	ENT↑	
089	+	
090	STO9	
091	RCLA	
092	ENT↑	
093	+	
094	GSBa	$k_4/2$
095	RCL0	
096	ENT↑	
097	+	
098	+	
099	RCLE	
100	+	
101	3	
102	÷	
103	RCLD	
104	+	
105	STOD	$y_i' + (k_1 + 2k_2 + 2k_3 + k_4)/6$
106	LSTX	
107	RCL0	
108	RCLE	
109	+	
110	3	
111	÷	$y_i' + (k_1 + k_2 + k_3)/6$
112	+	

REGISTERS

0	1	2	3	4	5	6	7	8	9
Used									Used
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
h/2	x_i	y_i	y_i', h	Used	

Step	Key	Comments
113	RCLA	
114	ENT↑	
115	+	h
116	STOE	
117	×	
118	RCLC	$y_i + h[y_i' + (k_1 + k_2 + k_3)/6]$
119	+	
120	STOC	
121	RCLE	
122	RCLB	$x_{i+1} = x_i + h$
123	+	
124	STOB	
125	GSB8	y_{i+1}
126	RCLC	
127	GSB8	
128	F0?	
129	SPC	Loop again.
130	GTOd	----------
131	*LBLa	Routine to find k_1, k_2, k_3,
132	RCL9	k_4
133	RCLD	
134	+	----------
135	RCL9	
136	*LBLb	
137	2	
138	÷	
139	RCLD	
140	+	
141	R↑	
142	×	
143	RCLC	
144	+	
145	RCLB	
146	R↑	
147	+	----------
148	*LBLE	User-defined f(x, y) or
149	RCLA	f(x, y, y′)
150	×	
151	RTN	----------
152	*LBL8	Output routine.
153	F0?	If F0 set, have AUTO
154	PRTX	mode: print/pause and
155	F0?	return.
156	RTN	If F0 clear, halt to display
157	R/S	result.
158	RTN	----------
159	*LBLe	AUTO toggle.
160	F0?	
161	GTO0	
162	SF0	
163	1	
164	RTN	
165	*LBL0	
166	CF0	
167	0	
168	RTN	

LABELS					FLAGS
A h	B $x_0 \uparrow y_0$	C y_0'	D →x_i; y_i	E f(x, y, y′)	0 AUTO
a Used	b Used	c	d 2nd-order	e AUTO?	1 2nd-order
0 Auto toggle	1	2	3	4	2
5	6	7	8 Output	9 1st-order	3

SET STATUS

FLAGS	ON	OFF	TRIG		DISP	
0	☐	☒	DEG	☒	FIX	☒
1	☐	☒	GRAD	☐	SCI	☐
2	☐	☒	RAD	☐	ENG	☐
3	☐	☒			n	2

L12-01

Interpolations

Step	Key	Comments
001	*LBLA	F1 set for linear.
002	SF1	y_0
003	STO7	
004	X⇄Y	x_0
005	STOA	
006	RTN	------------
007	*LBLB	
008	STO8	y_1
009	X⇄Y	
010	STOB	x_1
011	RTN	------------
012	*LBLC	
013	STO9	y_2
014	X⇄Y	
015	STOC	x_2
016	CF0	F0 clear first time through.
017	CF1	F1 clear for Lagrangian.
018	RTN	------------
019	*LBLD	
020	STO6	
021	F1?	If linear, GTO 1.
022	GTO1	
023	F0?	If second time through,
024	GTO0	GTO 0.
025	RCLA	---------
026	RCLB	Lagrangian, first time
027	-	through.
028	RCLA	
029	RCLC	
030	-	
031	x	
032	ST÷7	$y_0/[(x_0-x_1)(x_0-x_2)]$
033	RCLB	
034	RCLA	
035	-	
036	RCLB	
037	RCLC	
038	-	
039	x	
040	ST÷8	$y_1/[(x_1-x_0)(x_1-x_2)]$
041	RCLC	
042	RCLA	
043	-	
044	RCLC	
045	RCLB	
046	-	
047	x	
048	ST÷9	$y_2/[(x_2-x_0)(x_2-x_1)]$
049	SF0	
050	*LBL0	F0 set from now on.
051	RCL6	------------
052	RCLB	Lagrangian
053	-	
054	STOE	
055	RCL6	
056	RCLC	
057	-	
058	STOI	$(x-x_1)(x-x_2)$
059	x	
060	RCL7	$L_0(x)$
061	x	
062	RCL6	
063	RCLA	
064	-	
065	STOD	
066	RCLI	$(x-x_0)(x-x_2)$
067	x	
068	RCL8	
069	x	$L_1(x)$
070	+	
071	RCLD	
072	RCLE	
073	x	$(x-x_0)(x-x_1)$
074	RCL9	
075	x	$L_2(x)$
076	+	
077	PRTX	$P_2(x)$
078	RTN	------------
079	*LBL1	Linear interpolation.
080	RCLB	
081	RCL6	
082	-	
083	RCL7	
084	x	
085	RCL6	
086	RCLA	
087	-	
088	RCL8	
089	x	
090	+	
091	RCLB	$y = \frac{(x_1-x)y_1+(x-x_0)y_0}{x_1-x_0}$
092	RCLA	
093	-	
094	÷	
095	PRTX	
096	RTN	
097	*LBLa	
098	STOA	------------
099	RTN	Finite difference.
100	*LBLb	x_3
101	STOB	------------
102	RTN	h
103	*LBLc	------------
104	STOI	y_4
105	R↓	
106	STOE	
107	R↓	y_3
108	STOD	
109	R↑	y_2
110	-	
111	3	$-3y_3+3y_2$
112	x	

REGISTERS

0	1	2	3	4	5	6	7	8	9
						x	Used	Used	Used

S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
x_0, x_3	x_1, h	x_2, u	$x-x_0$, y_2, Used	$x-x_1$, y_3	$x-x_2$, y_4

Step	Key Entry	Comments
113	X⇄Y	$-3y_3 + 3y_2 - y_1$
114	-	y_4
115	R↑	$\delta^3 y_{-1/2}$
116	+	
117	STO9	
118	RCLE	
119	RCLD	
120	-	$\delta y_{-1/2}$
121	STO7	
122	RCLI	
123	RCLD	
124	+	
125	RCLE	
126	2	
127	×	
128	-	$\delta^2 y_0$
129	STO8	
130	RTN	------------
131	*LBLd	Compute y given x.
132	RCLA	
133	-	
134	RCLB	
135	÷	$u = (x - x_3)/h$
136	STOC	
137	RCL7	
138	×	
139	RCLE	
140	+	$y_3 + u\,\delta y_{-1/2}$
141	RCLC	
142	RCLC	
143	1	
144	+	
145	×	
146	STOD	$u(u+1)$
147	2	
148	÷	
149	RCL8	
150	×	$\frac{1}{2} u(u+1)\,\delta^2 y_0$
151	+	
152	RCLD	
153	RCLC	
154	1	
155	-	
156	×	
157	6	
158	÷	
159	RCL9	
160	×	$\frac{1}{6} u(u+1)(u-1)\,\delta^3 y_{-1/2}$
161	+	
162	PRTX	y
163	RTN	------------

LABELS

A $x_0 \uparrow y_0$	B $x_1 \uparrow y_1$	C $x_2 \uparrow y_2$	D $x \rightarrow y$	E
a x_3	b h	c $y_1 \uparrow y_2 \uparrow y_3 \uparrow y_4$	d $x \rightarrow y$	e
0 2nd time	1 Linear	2	3	4
5	6	7	8	9

FLAGS

0	2nd time
1	Linear
2	
3	

SET STATUS

FLAGS	ON	OFF	TRIG		DISP	
0	☐	☒	DEG	☒	FIX	☒
1	☐	☒	GRAD	☐	SCI	☐
2	☐	☒	RAD	☐	ENG	☐
3	☐	☒			n	2

Coordinate Transformations

Key	Comment
*LBLA	Store θ.
STO3	
CLX	
GSBa	Store 0, y_0, x_0.
RCL9	Set rotation axis of 0, 0, 1 in display and recall θ.
RCL9	
1	
RCL3	
*LBLb	
CF0	Store θ, a, b, and c.
STO3	
R↑	
STO0	
R↑	
STO1	
R↑	
STO2	
→P	Calculate $\sqrt{a^2 + b^2 + c^2}$
X⇄Y	
R↓	
→P	
ST÷0	Calculate unit vector components.
ST÷1	
ST÷2	
RTN	
*LBLa	Store z_0, y_0, x_0.
CF0	
STO9	
R↓	
STO8	
R↓	
STO7	
RTN	
*LBLC	
SF2	Set 2 –D flag and input dummy zero.
0	
*LBLc	
RCL9	Store $(x - x_0)$, $(y - y_0)$, $(z - z_0)$.
-	
STOC	
CLX	
RCL8	
-	
STOB	
CLX	
RCL7	
-	
STOA	
F0?	
GTO0	Calculate matrix coefficients if not already done.
GSB5	
*LBL0	
CF0	Calculate x or x′ coefficient.
SPC	
1	
1	
STOI	
GSB1	
STO4	
1	Calculate y or y′ coefficient.
2	
STOI	
GSB1	Stop if 2 –D.
STO5	
F2?	
RTN	
1	Calculate z or z′ coefficient.
3	
STOI	
GSB1	
STO6	
0	Stop and display final 3 –D results.
RCL4	
RCL5	
RCL6	
RTN	
*LBLc	Set matrix done flag. (Program pointer says matrix done.)
SF0	
GTOc	
*LBLC	
SF0	Set matrix done flag.
GTOC	
*LBL1	Matrix multiply x and y.
0	
RCLA	
GSB4	
RCLB	
GSB4	
RCLC	
P⇄S	Fetch x_0, y_0 or z_0 depending on which coefficient is being calculated.
RCLi	
P⇄S	
STOD	
R↓	Matrix multiply z.
GSB4	If calculating P from P′ add appropriate translation distance.
RCLD	
X⇄Y	
F1?	
+	
PRTX	Output result, set stack for 2 –D stop.
RCL4	
X⇄Y	
RTN	
*LBL4	Matrix multiply and counter.
RCLi	
x	
+	
ISZI	
ISZI	
ISZI	
RTN	
*LBL5	

REGISTERS

0 a	1 b	2 c	3 θ	4 x(x′)	5 y(y′)	6 z(z′)	7 x_0	8 y_0	9 z_0
S0	S1 ℓ_1	S2 ℓ_2	S3 ℓ_3	S4 m_1	S5 m_2	S6 m_3	S7 n_1	S8 n_2	S9 n_3

A x(x′)	B y(y′)	C z(z′)	D $\cos\theta$, (x_0, y_0, z_0)	E $\sin\theta$	I control

Step	Key	Comment
113	RCL3	Calculate sin θ, cos θ and 1 − cos θ.
114	1	
115	→R	
116	STOD	
117	CHS	
118	X⇄Y	
119	STOE	
120	CLX	
121	1	
122	+	
123	RCL0	Recall unit vector components.
124	RCL1	
125	RCL2	
126	R↑	1 − cos θ
127	P⇄S	Store 1 − cos θ in $R_{S1} - R_{S9}$.
128	STO1	
129	STO2	
130	STO3	
131	STO4	
132	STO5	
133	STO6	
134	STO7	
135	STO8	
136	STO9	
137	R↓	Multiply by c unit vector component.
138	ST×7	
139	ST×8	
140	ST×9	
141	ST×9	
142	ST×6	
143	ST×3	
144	R↓	
145	ST×5	Multiply by b unit vector component.
146	ST×5	
147	ST×2	
148	ST×8	
149	ST×4	
150	ST×6	
151	R↓	
152	ST×2	Multiply by a unit vector component.
153	ST×1	
154	ST×1	
155	ST×4	
156	ST×7	
157	ST×3	
158	R↓	
159	R↓	c→X
160	RCLD	
161	ST+1	Add cos θ.
162	ST+5	
163	ST+9	
164	CLX	
165	RCLE	
166	F1?	sin θ
167	CHS	CHS for P′→P.
168	×	
169	ST−2	c sin θ
170	ST+4	
171	CLX	
172	LSTX	
173	×	b sin θ
174	ST+3	
175	ST−7	
176	CLX	
177	LSTX	
178	×	a sin θ
179	ST+8	
180	ST−6	
181	P⇄S	Matrix complete.
182	RTN	
183	*LBLE	Set 2D flag and input dummy zero.
184	SF2	
185	0	
186	*LBLe	Input x, y, z.
187	STOC	
188	R↓	
189	STOB	
190	R↓	
191	STOA	
192	SF1	
193	F0?	Set inverse flag.
194	GTO6	Calculate matrix coefficients if not previously done.
195	GSB5	
196	*LBL6	
197	GSB0	P′→P
198	CF1	
199	RTN	
200	*LBLe	
201	SF0	Set matrix done flag.
202	GTOe	
203	*LBLE	
204	SF0	Set matrix done flag.
205	GTOE	

LABELS					FLAGS
A x_0↑y_0↑θ	B	C x↑y→P′	D	E x′↑y′→P	0 Matrix done
a x_0↑y_0↑z_0	b a↑b↑c↑θ	c x↑y↑z→P′	d	e x′↑y′↑z′→P	1 P′→P
0 Used	1 Mult.	2	3	4 Mult.	2 2D
5 Matrix	6	7	8	9	3

SET STATUS

FLAGS	ON	OFF	TRIG		DISP	
0	☐	☒	DEG	☒	FIX	☒
1	☐	☒	GRAD	☐	SCI	☐
2	☐	☒	RAD	☐	ENG	☐
3	☐	☒			n	2

Intersections

Step	Key	Comments
001	*LBLa	Input P_1, P_1'.
002	GSB5	
003	*LBLA	Input P_1 and θ_1.
004	1	
005	STOE	
006	R↓	
007	STOD	
008	R↓	
009	STOC	
010	R↓	
011	STOB	
012	RTN	
013	*LBLb	Input P_2, P_2'.
014	GSB5	
015	*LBLB	Input P_2 and θ_2.
016	2	
017	STOI	
018	R↓	
019	STOA	
020	R↓	
021	STO9	
022	R↓	
023	STO8	
024	RTN	
025	*LBLD	Input x_{01}, y_{01}, r_1.
026	3	
027	STOI	
028	R↓	
029	STOA	
030	R↓	
031	STO9	
032	R↓	
033	STO8	
034	RTN	
035	*LBLE	Input x_{02}, y_{02}, r_2.
036	4	
037	STOE	
038	R↓	
039	STOD	
040	R↓	
041	STOC	
042	R↓	
043	STOB	
044	RTN	
045	*LBL5	
046	STO7	Transform P and P' to P – θ form.
047	X⇄Y	
048	STO6	
049	R↓	
050	-	
051	X⇄Y	
052	R↑	
053	-	
054	→P	
055	R↓	
056	+	

Step	Key	Comments
057	RCL6	
058	RCL7	
059	LSTX	
060	RTN	
061	*LBLe	Set flag for alternate point solution.
062	SF1	
063	GTO0	
064	*LBLd	Start primary point solution.
065	CF1	
066	*LBL0	Select type of solution.
067	CF2	
068	SPC	
069	GTOi	Line-line intersect.
070	*LBL2	
071	RCLE	
072	1	Check for input error.
073	X≠Y?	
074	GTO6	
075	RCLD	If $\theta_1 = \pm 90°$ go to special solution 8.
076	COS	
077	X=0?	
078	GTO8	
079	RCLA	If $\theta_2 = \pm 90°$ go to special solution 7.
080	COS	
081	X=0?	
082	GTO7	
083	RCLB	
084	RCLD	
085	TAN	Calculate x_p.
086	STO6	
087	x	
088	RCL8	
089	RCLA	
090	TAN	
091	STO7	
092	x	
093	-	
094	RCL9	
095	+	
096	RCLC	
097	-	
098	RCL6	
099	RCL7	
100	-	
101	÷	
102	*LBL9	
103	ENT↑	
104	ENT↑	Calculate y_p.
105	PRTX	
106	RCLB	
107	-	
108	RCL6	
109	x	
110	RCLC	
111	+	
112	PRTX	

REGISTERS

0	1	2	3	4	5	6	7	8	9
						Used	Used	x_2, x_{c1}	y_2, y_{c1}

S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
θ_2, r_1	x_1, x_{c2}	y_1, y_{c2}	θ_1, r_2	Code	Code

Step	Key	Comment
113	RTN	Calculate x_p for $\theta_1 = \pm 90°$.
114	*LBL7	
115	RCLD	
116	TAN	
117	STO6	
118	RCL8	------------
119	GTO9	Calculate x_p, y_p for $\theta_2 = \pm 90°$.
120	*LBL8	
121	RCLB	
122	RCLB	
123	PRTX	
124	RCL8	
125	-	
126	RCLA	
127	TAN	
128	x	
129	RCL9	
130	+	
131	PRTX	
132	RTN	------------
133	*LBL3	Circle-line or circle-circle.
134	RCLE	------------
135	X⇄I	
136	GTOi	
137	*LBL1	
138	GSB0	Calculate x_p for circle-line.
139	-	
140	STO7	
141	X⇄Y	
142	RCLD	
143	-	
144	COS	
145	RCL6	
146	x	
147	CHS	
148	X<0?	
149	SF2	
150	ENT↑	
151	X²	
152	RCL7	
153	-	
154	√X	
155	F2?	
156	CHS	
157	+	Calculate P_2.
158	÷	Calculate P_1.
159	F1?	Bring P_2 back if flag is set.
160	LSTX	Calculate x_p.
161	RCLD	
162	X⇄Y	
163	→R	
164	RCLB	
165	+	
166	PRTX	
167	X⇄Y	
168	RCLC	Calculate y_p.
169	+	
170	PRTX	
171	RTN	
172	*LBL4	Calculate x_p for circle-circle.
173	GSB0	
174	+	
175	RCLD	
176	X²	
177	-	
178	2	
179	÷	
180	RCL6	
181	÷	
182	RCLA	
183	÷	
184	COS⁻¹	
185	F1?	
186	CHS	
187	+	
188	RCLA	
189	→R	
190	RCL8	
191	+	Calculate y_p for circle-circle.
192	PRTX	
193	X⇄Y	
194	RCL9	
195	+	
196	PRTX	
197	RTN	
198	*LBL0	Calculate α and D.
199	X⇄I	
200	RCLC	
201	RCL9	
202	-	
203	RCLB	
204	RCL8	
205	-	
206	→P	
207	STO6	
208	X²	Calculate D^2 and r^2.
209	RCLA	
210	X²	
211	RTN	------------

LABELS

A $x_1 \uparrow y_1 \uparrow \theta_1$	B $x_2 \uparrow y_2 \uparrow \theta_2$	C	D $x_{01} \uparrow y_{01} \uparrow r_1$	E $x_{02} \uparrow y_{02} \uparrow r_2$
a $x_1 \uparrow y_1 \uparrow x' \uparrow y_1'$	b $x_2 \uparrow y_2 \uparrow x_2' \uparrow y_2'$	c	d $\rightarrow x_{01}, y_{01}$	e $\rightarrow x_{02}, y_{02}$
0 Used	1 x_p	2 Line-line	3 Used	4 x_p
5 Line spec.	6	7 Vert. line	8 Vert. line	9 y_p

FLAGS

0	
1	Prim/sec
2	Quad
3	

SET STATUS

FLAGS	ON	OFF	TRIG		DISP	
0	☐	☒	DEG	☒	FIX	☒
1	☐	☒	GRAD	☐	SCI	☐
2	☐	☒	RAD	☐	ENG	☐
3	☐	☒			n 2	

Circles

Step	Key	Comment
001	*LBLa	Store x_1, y_1.
002	STO4	
003	R↓	
004	STO3	
005	RTN	------------
006	*LBLb	
007	STO6	Store x_2, y_2.
008	R↓	
009	STO5	
010	RTN	------------
011	*LBLc	
012	STO8	
013	R↓	Store x_3, y_3.
014	STO7	
015	RTN	------------
016	*LBLd	If $x_1 = x_2$ or
017	SPC	$x_1 = x_3$ then
018	RCL5	
019	RCL3	P_1
020	X=Y?	↙ ↖
021	GTO0	$P_3 \rightarrow P_2$
022	RCL7	
023	RCL3	
024	X=Y?	
025	GTO0	
026	*LBL1	
027	RCL6	Calculate k_1 and N_1.
028	RCL4	
029	-	
030	STO1	
031	RCL6	
032	RCL4	
033	+	
034	x	
035	RCL5	
036	RCL3	
037	+	
038	RCL5	
039	RCL3	
040	-	
041	ST÷1	
042	STOC	
043	x	
044	+	
045	RCLC	
046	÷	
047	2	
048	÷	
049	STOC	
050	RCL8	
051	RCL4	
052	-	Calculate k_2 and N_2.
053	STO2	
054	RCL8	
055	RCL4	
056	+	

Step	Key	Comment
057	x	
058	RCL7	
059	RCL3	
060	-	
061	ST÷2	
062	STOD	
063	RCL7	
064	RCL3	
065	+	
066	x	
067	+	
068	2	
069	÷	
070	RCLD	
071	÷	
072	STOE	
073	RCLC	$y_0 = \frac{k_2 - k_1}{N_2 - N_1}$
074	-	
075	RCL2	
076	RCL1	
077	-	
078	÷	
079	STOD	
080	RCLE	
081	X⇄Y	
082	RCL2	
083	x	$x_0 = k_2 - N_2 y_0$
084	-	
085	STOC	Output x.
086	PRTX	
087	RCL7	
088	-	Output y.
089	RCLD	
090	PRTX	
091	RCL8	
092	-	Output r and stop with x in Z, y in Y and r in X.
093	→P	
094	RCLD	
095	X⇄Y	
096	RCLC	
097	X⇄Y	
098	STOE	
099	SPC	
100	PRTX	
101	RTN	
102	*LBL0	------------
103	RCL7	
104	RCL8	P_1
105	RCL3	↙ ↖
106	STO7	$P_3 \rightarrow P_2$
107	CLX	
108	RCL5	
109	STO3	
110	CLX	
111	RCL4	
112	STO8	

REGISTERS

0	1	2	3	4	5	6	7	8	9
	N_1, i	N_2, θ_0	x_1	y_1	x_2	y_2	x_3	y_3	θ
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
$\Delta\theta$	n	x_c, k_1	y_c	r	n − i

Step	Key	Comment
113	CLX	
114	RCL6	
115	STO4	
116	R↓	
117	STO6	
118	R↓	
119	STO5	
120	GTO1	
121	*LBLA	Input x_0, y_0, and r.
122	STOE	
123	R↓	
124	STOD	
125	R↓	
126	STOC	
127	RTN	
128	*LBLC	Calculate n from $\Delta\theta$.
129	1	
130	CHS	
131	COS⁻¹	
132	ENT↑	
133	+	
134	X⇄Y	
135	÷	
136	LSTX	
137	GTO0	
138	*LBLD	Calculate $\Delta\theta$ from n.
139	1	
140	CHS	
141	COS⁻¹	
142	ENT↑	
143	+	
144	X⇄Y	
145	÷	
146	LSTX	
147	X⇄Y	
148	*LBL0	Store $\Delta\theta$, n, θ_0 and $\theta_0 - \Delta\theta$.
149	STOA	
150	R↓	
151	ABS	
152	.	
153	5	
154	+	
155	INT	
156	STOB	
157	STOI	
158	R↓	
159	STO2	
160	STO9	
161	RCLA	
162	ST-9	
163	RTN	
164	*LBLE	Calculate and output θ, i, x_i and y_i.
165	SPC	
166	RCLA	
167	ST+9	
168	RCL9	
169	PRTX	
170	DSZI	
171	*LBL8	
172	RCLB	
173	RCLI	
174	-	
175	PRTX	
176	STO1	
177	GSB2	
178	R↑	
179	CLX	
180	RCL9	
181	R↑	
182	CLX	
183	RCL1	
184	R↑	
185	R↑	
186	RTN	
187	*LBLe	Set to begin automatic loop.
188	RCLB	
189	STOI	
190	RCL2	
191	RCLA	
192	-	
193	STO9	
194	*LBL4	Automatic loop.
195	GSBE	
196	ISZI	
197	*LBL8	
198	DSZI	
199	GTO4	
200	RTN	
201	*LBLB	Calculate x and y for given θ.
202	SPC	
203	GTO8	
204	*LBL2	
205	RCL9	Calculate x and y for contents of R_9.
206	*LBL8	
207	RCLE	
208	→R	
209	RCLC	
210	+	
211	PRTX	
212	X⇄Y	
213	RCLD	
214	+	
215	PRTX	
216	RTN	

LABELS

A $x_0 \uparrow y_0 \uparrow r$	B $\theta \rightarrow x, y$	C $\theta_0 \uparrow \Delta\theta$	D $\theta_0 \uparrow n$	E $\rightarrow \theta$, i, x, y
a $x_1 \uparrow y_1$	b $x_2 \uparrow y_2$	c $x_3 \uparrow y_3$	d $\rightarrow x_0, y_0, r$	e AUTO
0 Used	1 $\rightarrow x_0, y_0, r$	2 $\rightarrow x_i, y_i$	3	4 Loop
5	6	7	8	9

FLAGS

0	
1	
2	
3	

SET STATUS

FLAGS	ON	OFF	TRIG		DISP	
0	☐	☒	DEG	☒	FIX	☒
1	☐	☒	GRAD	☐	SCI	☐
2	☐	☒	RAD	☐	ENG	☐
3	☐	☒			n	2

Spherical Triangles

Step	Key	Comments
001	*LBLE	ASA (AcB)
002	SF1	
003	*LBLB	SAS (aCb)
004	STOB	Second S (or A)
005	R↓	
006	STOC	Angle (or S)
007	R↓	
008	STOA	First S (or A)
009	RCLC	
010	COS	
011	RCLA	
012	SIN	$\cos c = \cos a \cos b + \sin a \sin b \cos C$
013	×	
014	RCLB	
015	SIN	
016	×	
017	RCLA	
018	COS	$\cos C = -\cos A \cos B + \sin A \sin B \cos c$
019	RCLB	
020	COS	
021	×	
022	F1?	
023	CHS	
024	+	
025	COS⁻¹	
026	STOC	Third S (or A)
027	GTO8	
028	*LBLD	AAA (ABC)
029	SF1	
030	*LBLA	SSS (abc)
031	STOC	
032	R↓	
033	STOB	Store three sides (or angles).
034	R↓	
035	STOA	
036	*LBL0	
037	GSBb	LBL b finds one angle (or side)
038	STOE	
039	GSBb	
040	STOI	
041	GSBb	
042	STOD	
043	CF1	
044	*LBL9	
045	RCLA	Output routine. First side (or angle)
046	GSB8	
047	RCLD	First angle (or side)
048	GSB8	
049	RCLB	Second side (or angle)
050	GSB8	
051	RCLE	Second angle (or side)
052	GSB8	
053	RCLC	Third side (or angle)
054	GSB8	
055	RCLI	Third angle (or side)
056	GSB8	
057	F0?	
058	SPC	
059	RTN	
060	*LBLb	Subroutine to find one angle (side).
061	RCLA	
062	RCLB	
063	STOA	
064	RCLC	Rotate sides (angles)
065	STOB	
066	R↓	
067	R↓	
068	STOC	$\cos A = \frac{\cos a - \cos b \cos c}{\sin b \sin c}$
069	COS	
070	RCLA	
071	COS	
072	RCLB	
073	COS	
074	×	$\cos a = \frac{\cos A + \cos B \cos C}{\sin B \sin C}$
075	F1?	
076	CHS	
077	−	
078	RCLA	
079	SIN	
080	÷	
081	RCLB	
082	SIN	
083	÷	
084	COS⁻¹	
085	RTN	
086	*LBL8	AUTO output.
087	F0?	Print if F0 set.
088	PRTX	
089	F0?	
090	RTN	
091	R/S	Else halt.
092	RTN	
093	*LBLa	
094	SF1	AAS (A,B,a) Ambiguous cases.
095	*LBLC	SSA (a, b, A)
096	STOE	Angle (or side)
097	R↓	
098	STOB	Second side (or angle)
099	SIN	
100	X⇄Y	
101	STOA	First side (or angle)
102	SIN	
103	÷	
104	RCLE	$\sin B = \frac{\sin b \sin A}{\sin a}$
105	SIN	
106	×	
107	SIN⁻¹	$\sin b = \frac{\sin B \sin a}{\sin A}$
108	GSBd	
109	RCLA	
110	RCLB	Find one solution.
111	X>Y?	If a<b (A<B), have 2 solutions.
112	GTO0	

REGISTERS

0	1	2	3	4	5	6	7	8	9
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
First S (or A)	Second S (or A)	Third S (or A)	First A (or S)	Second A (or S)	Third A (or S)

Step	Key	Comment
113	CLX	Else end.
114	RTN	----------
115	*LBL0	For 2nd solution, $B \leftarrow \cos^{-1}(-\cos B)$
116	RCLI	
117	COS	
118	CHS	
119	COS⁻¹	----------
120	*LBLd	Routine finds one solution given 2 angles and 2 sides.
121	STOI	
122	RCLE	
123	+	
124	2	
125	÷	$\tan\frac{c}{2} = \dfrac{\sin\left(\frac{A+B}{2}\right)\tan\left(\frac{a-b}{2}\right)}{\sin\left(\frac{A-B}{2}\right)}$
126	ENT↑	
127	SIN	
128	X⇄Y	
129	RCLI	
130	-	$\cot\frac{C}{2} = \dfrac{\sin\left(\frac{a+b}{2}\right)\tan\left(\frac{A-B}{2}\right)}{\sin\left(\frac{a-b}{2}\right)}$
131	SIN	
132	÷	
133	RCLA	
134	RCLB	
135	-	
136	2	
137	÷	
138	TAN	
139	×	
140	F1?	
141	1/X	
142	TAN⁻¹	
143	ENT↑	
144	+	
145	STOC	
146	COS	
147	RCLA	
148	COS	$\cos C = \dfrac{\cos c - \cos a \cos b}{\sin a \sin b}$
149	RCLB	
150	COS	
151	×	
152	F1?	$\cos c = \dfrac{\cos C + \cos A \cos B}{\sin A \sin B}$
153	CHS	
154	-	
155	RCLA	
156	SIN	
157	÷	
158	RCLB	
159	SIN	
160	÷	
161	COS⁻¹	
162	STOD	
163	GSB9	
164	CLX	
165	RTN	----------
166	*LBLe	AUTO toggle.
167	F0?	
168	GTO0	
169	SF0	
170	1	
171	RTN	
172	*LBL0	
173	CF0	
174	0	
175	RTN	

LABELS

A SSS	B SAS	C SSA	D AAA	E ASA
a AAS	b	c	d	e Auto?
0 Used	1	2	3	4
5	6	7	8 Auto out	9 Output

FLAGS

0	Auto
1	Angles
2	
3	

SET STATUS

FLAGS	ON	OFF	TRIG		DISP	
0	☐	☒	DEG	☒	FIX	☒
1	☐	☒	GRAD	☐	SCI	☐
2	☐	☒	RAD	☐	ENG	☐
3	☐	☒			n	2

L17-01

Gamma Function

Step	Key	Comments
001	*LBLA	Load constants.
002	P⇄S	
003	.	
004	5	
005	7	
006	7	
007	1	
008	9	
009	1	
010	6	
011	5	
012	2	
013	CHS	b_1
014	STO1	
015	.	
016	9	
017	8	
018	8	
019	2	
020	0	
021	5	
022	8	
023	9	
024	1	
025	STO2	b_2
026	.	
027	8	
028	9	
029	7	
030	0	
031	5	
032	6	
033	9	
034	3	
035	7	
036	CHS	
037	STO3	b_3
038	.	
039	9	
040	1	
041	8	
042	2	
043	0	
044	6	
045	8	
046	5	
047	7	
048	STO4	
049	.	b_4
050	7	
051	5	
052	6	
053	7	
054	0	
055	4	
056	0	
057	7	
058	8	
059	CHS	b_5
060	STO5	
061	.	
062	4	
063	8	
064	2	
065	1	
066	9	
067	9	
068	3	
069	9	
070	4	
071	STO6	b_6
072	.	
073	1	
074	9	
075	3	
076	5	
077	2	
078	7	
079	8	
080	1	
081	8	
082	CHS	b_7
083	STO7	
084	.	
085	0	
086	3	
087	5	
088	8	
089	6	
090	8	
091	3	
092	4	
093	3	
094	STO8	b_8
095	CLX	
096	P⇄S	
097	RTN	
098	*LBLB	
099	P⇄S	$x \rightarrow \Gamma(x)$
100	1	
101	-	$(x - 1)$
102	X<0?	
103	GTOE	Error if $(x - 1) < 0$.
104	INT	
105	LSTX	If $(x-1)$ integer, GTO b.
106	X=Y?	
107	GTOb	
108	1	
109	STO9	
110	X⇄Y	
111	*LBL9	Exit when < 1.
112	X≰Y?	

REGISTERS

0	1	2	3	4	5	6	7	8	9

S0	S1	S2	S3	S4	S5	S6	S7	S8	S9
	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	Π

A	B	C	D	E	I

Step	Key	Comments
113	GTO0	R_9 accumulates product
114	ST×9	$(x-1)(x-2)(x-3)$...
115	1	
116	-	--------
117	GTO9	Polynomial approx.
118	*LBL0	Here $0 <$ argument ≤ 1.
119	ENT↑	
120	ENT↑	
121	ENT↑	
122	RCL8	
123	×	
124	RCL7	
125	+	
126	×	
127	RCL6	
128	+	
129	×	
130	RCL5	
131	+	
132	×	
133	RCL4	
134	+	
135	×	
136	RCL3	
137	+	
138	×	
139	RCL2	
140	+	
141	×	
142	RCL1	
143	+	
144	×	
145	1	
146	+	
147	RCL9	
148	×	
149	PRTX	$\Gamma(x)$
150	P⇄S	
151	RTN	
152	*LBLb	------------
153	N!	If $(x-1)$ integer, simply
154	PRTX	take factorial.
155	P⇄S	
156	RTN	------------

LABELS

A START	B $x \rightarrow \Gamma(x)$	C	D	E
a	b Integers	c	d	e
0 Approx.	1	2	3	4
5	6	7	8	9 Π loop

FLAGS

0	
1	
2	
3	

SET STATUS

FLAGS	ON	OFF
0	☐	☒
1	☐	☒
2	☐	☒
3	☐	☒

TRIG	
DEG	☒
GRAD	☐
RAD	☐

DISP	
FIX	☒
SCI	☐
ENG	☐
n	2

Bessel Functions, Error Function

Step	Key	Comments
001	*LBLA	Bessel n
002	STOA	
003	RTN	
004	*LBLB	$J_n(x)$
005	GSBa	Initialize
006	SF0	F0 set for J_n.
007	*LBL9	
008	GSBb	Main summing loop.
009	CF2	Compute term (even k).
010	ST+9	
011	GSBb	Accumulate even terms.
012	F2?	Compute term (odd k).
013	GTO9	F2 clear for last term.
014	RCLC	Loop again.
015	RCL9	
016	ENT↑	$R_C = T_n$, $R_E = T_0$, at end.
017	+	
018	RCLE	$J_n(x) = \frac{T_n}{-T_0 + 2\sum_0 T_k}$
019	-	
020	÷	
021	PRTX	
022	SPC	
023	RTN	
024	*LBLa	Initialization for Bessel
025	1	(J_n and I_n).
026	.	
027	5	
028	×	
029	STOC	
030	RCLA	$R_C \leftarrow 1.5x$
031	X≤Y?	n
032	X⇄Y	
033	6	max (n, 1.5x)
034	+	
035	RCLC	
036	9	
037	×	
038	RCLC	
039	2	
040	+	Compute m.
041	÷	
042	+	
043	2	
044	÷	
045	INT	
046	ENT↑	
047	+	
048	2	
049	+	
050	STOI	
051	3	
052	RCLC	
053	÷	$I \leftarrow m + 2$
054	STOB	
055	8	
056	STOE	$R_B \leftarrow 2/x$
057	STO9	
058	EEX	
059	CHS	
060	9	
061	STOD	
062	RTN	
063	*LBLb	
064	DSZI	Compute one term.
065	SF2	F2 set except for k = 0.
066	RCLI	
067	RCLA	
068	X≠Y?	
069	GTO0	
070	RCLD	If k = n, save $T_n \rightarrow R_C$.
071	STOC	
072	*LBL0	
073	R↓	
074	RCLE	T_{k+1}
075	F0?	
076	CHS	
077	X⇄Y	CHS for J_n $(-T_{k+1})$
078	RCLB	
079	×	$\frac{2k}{x}$
080	RCLD	
081	STOE	$T_{k+1} \leftarrow T_k$
082	×	
083	+	$T_k \leftarrow \frac{2k}{x} T_k \pm T_{k+1}$
084	STOD	
085	RTN	
086	*LBLC	Compute $J_0(x)$, $J_1(x)$
087	RCLE	
088	RCL9	
089	ENT↑	
090	+	
091	RCLE	
092	-	
093	÷	
094	R/S	$J_0(x) = \frac{T_0}{-T_0 + 2\sum_0 T_k}$
095	RCLD	
096	CHS	
097	RCL9	
098	ENT↑	
099	+	
100	RCLE	
101	-	
102	÷	
103	RTN	
104	*LBLD	
105	CF0	$I_n(x)$
106	GSBa	F0 clear for $I_n(x)$.
107	*LBL8	Initialize.
108	ST+9	
109	GSBb	Accumulate each term.
110	F2?	Compute next term.
111	GTO8	F2 clear for last term.
112	RCLC	

REGISTERS

0	1	2	3	4	5	6	7	8	9 ΣT_k
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
n; erf term	2/x;	1.5x, T_n,	T_k; $(e^{x^2}\sqrt{\pi})^{-1}$	T_{k+1};	k; places

Step	Key	Comments
113	RCL9	
114	ENT↑	$R_E = T_0$ at end.
115	+	
116	RCLE	
117	−	
118	÷	
119	2	$I_n(x) = e^x \dfrac{T_n}{-T_0 + 2 \sum_0 T_i}$
120	RCLB	
121	÷	
122	e^x	
123	x	
124	PRTX	
125	SPC	
126	RTN	------------
127	*LBLE	Erf and erfc
128	STOA	Initialization
129	X^2	
130	STOD	
131	2	
132	x	$2x^2 \rightarrow R_B$
133	STOB	
134	1	
135	STOC	
136	RCLD	
137	e^x	e^{x^2}
138	Pi	
139	√X	
140	x	$(e^{x^2} \sqrt{\pi})$
141	STOD	
142	3	
143	RCLA	
144	X>Y?	
145	GTO3	If $x > 3$, compute erfc first.
146	*LBL7	--------
147	RCLB	Loop for erf
148	RCLC	(Stack contains Σ_{n-1})
149	2	Let δ_n, Σ_n be n^{th} term and
150	+	sum thru 1^{st} n terms, resp.
151	STOC	$2n + 1$.
152	÷	
153	RCLA	
154	x	$\delta_n \leftarrow \delta_{n-1} \dfrac{2x^2}{2n+1}$
155	STOA	
156	+	$\Sigma_n \leftarrow \Sigma_{n-1} + \delta_n$
157	X⇄Y	
158	RND	RND (Σ_{n-1})
159	X⇄Y	
160	RND	RND (Σ_n)
161	X=Y?	If last 2 sums are equal to
162	GTO0	N places, exit.
163	LSTX	Σ_n
164	GTO7	--------
165	*LBL0	Exit erf.
166	LSTX	
167	RCLD	$\Sigma_n / (e^{x^2} \sqrt{\pi})$
168	÷	

Step	Key	Comments
169	2	erf(x)
170	x	
171	1	erfc = 1 − erf
172	X⇄Y	
173	−	
174	LSTX	
175	GTO2	--------
176	*LBL3	Find erfc, $x > 3$
177	RCLB	$1/(2x^2) \rightarrow R_B$
178	1/X	
179	STOB	
180	RCLA	$1/x \rightarrow R_A$
181	1/X	
182	STOA	--------
183	*LBL6	Loop for erfc
184	RCLB	(Stack contains Σ_{n-1})
185	RCLC	
186	2	
187	−	
188	STOC	$\delta_n \leftarrow -\delta_{n-1} \dfrac{2n-1}{2x^2}$
189	x	
190	RCLA	
191	x	
192	STOA	
193	+	$\Sigma_n \leftarrow \Sigma_{n-1} + \delta_n$
194	X⇄Y	
195	RND	RND (Σ_{n-1})
196	X⇄Y	
197	RND	RND (Σ_n)
198	X=Y?	If last 2 sums are equal
199	GTO8	to N places, exit.
200	LSTX	Σ_n
201	GTO6	--------
202	*LBL8	
203	LSTX	
204	RCLD	
205	÷	
206	1	erfc(x)
207	X⇄Y	
208	−	erf(x)
209	LSTX	
210	X⇄Y	
211	*LBL2	
212	PRTX	erf(x)
213	X⇄Y	
214	PRTX	erfc(x)
215	X⇄Y	
216	SPC	
217	RTN	------------
218	*LBLe	Set DSP to places of
219	STOI	accuracy desired for erf
220	DSPi	(1–9).
221	RTN	------------

LABELS

A n	B $x \rightarrow J_n(x)$	C J_0 ; J_1	D $x \rightarrow I_n(x)$	E x→erf, erfc
a Bes. init.	b One term	c	d	e Accuracy
0 Used	1	2 Print erf	3 x>3 (erfc)	4
5	6 erfc loop	7 erf loop	8 I_n loop	9 J_n loop

FLAGS

0	J_n
1	
2	k = 0 (Bessel)
3	

SET STATUS

FLAGS	ON	OFF	TRIG		DISP	
0	☐	☒	DEG	☒	FIX	☒
1	☐	☒	GRAD	☐	SCI	☐
2	☐	☒	RAD	☐	ENG	☐
3	☐	☒			n	2

L19-01

Hyperbolics

Key	Comments
*LBLA	Arc
SF2	Set F2 for inverse.
RTN	
*LBLB	Sinh
P⇄S	
R↑	
STO0	Save t
R↓	
F2?	If inverse, GTO 0.
GTO0	
e^x	
ENT↑	
1/X	Compute sinh x.
-	
2	
÷	
GTO1	Exit.
*LBL0	
STO9	
X^2	Compute $\sinh^{-1}$ x.
1	
+	
√X	
RCL9	
+	
LN	
*LBL1	
RCL0	Restore t.
R↓	
P⇄S	
RTN	
*LBLC	Cosh
P⇄S	
R↑	
STO0	Save t.
R↓	
F2?	If inverse, GTO 2.
GTO2	
e^x	
ENT↑	
1/X	
+	Compute cosh x.
2	
÷	
GTO3	Exit.
*LBL2	
STO9	
X^2	
1	Compute $\cosh^{-1}$ x.
-	
√X	
RCL9	
+	
LN	
*LBL3	Restore t.
RCL0	
R↓	
P⇄S	
RTN	
*LBLD	Tanh
P⇄S	
R↑	
STO0	Save t
R↑	
STO8	Save z
R↓	
R↓	
F2?	
GTO4	If inverse, GTO 4.
e^x	
ENT↑	
1/X	
-	
STO9	Compute tanh x.
LSTX	
ENT↑	
+	
+	
RCL9	
X⇄Y	
÷	Exit.
GTO5	
*LBL4	
ENT↑	
ENT↑	
1	
+	Compute $\tanh^{-1}$ x.
X⇄Y	
CHS	
1	
+	
÷	
LN	
2	
÷	
*LBL5	
RCL0	
RCL8	Restore t and z.
R↓	
R↓	
P⇄S	
RTN	
*LBLE	csch
F2?	
GTO6	
GSBB	
1/X	
RTN	csch x = $(\sinh x)^{-1}$
*LBL6	
SF2	csch^{-1} x = $\sinh^{-1}$ (1/x)
1/X	
GTOB	

REGISTERS

0	1	2	3	4	5	6	7	8	9

S0	S1	S2	S3	S4	S5	S6	S7	S8	S9
Save t								Save z	Used

A	B	C	D	E	I

```
113  *LBLc
114    F2?
115   GTO7
116   GSBC
117    1/X
118    RTN
119  *LBL7
120    SF2
121    1/X
122   GTOC
123  *LBLd
124    F2?
125   GTO8
126   GSBD
127    1/X
128    RTN
129  *LBL8
130    SF2
131    1/X
132   GTOD
```

Sech

sech x = $(\cosh x)^{-1}$

$\text{sech}^{-1} x = \cosh^{-1} (1/x)$

coth

coth x = $(\tanh x)^{-1}$

$\coth^{-1} x = \tanh^{-1} (1/x)$

LABELS					FLAGS
A Arc	B Sinh	C Cosh	D Tanh	E	0
a	b Csch	c Sech	d Coth	e	1
0 $\sinh^{-1}$	1 Exit sinh	2 $\cosh^{-1}$	3 Exit cosh	4 $\tanh^{-1}$	2 Arc
5 Exit tanh	6 csch^{-1}	7 sech^{-1}	8 $\coth^{-1}$	9	3

SET STATUS

FLAGS	ON	OFF	TRIG		DISP	
0	☐	☒	DEG	☒	FIX	☒
1	☐	☒	GRAD	☐	SCI	☐
2	☐	☒	RAD	☐	ENG	☐
3	☐	☒			n	2

Appendix A
MAGNETIC CARD SYMBOLS AND CONVENTIONS

SYMBOL OR CONVENTION	INDICATED MEANING
White mnemonic: x A	White mnemonics are associated with the user-definable key they are above when the card is inserted in the calculator's window slot. In this case the value of x could be input by keying it in and pressing A.
Gold mnemonic: y x f E	Gold mnemonics are similar to white mnemonics except that the gold f key must be pressed before the user-definable key. In this case y could be input by pressing f E.
x ↑ y A	↑ is the symbol for ENTER↑. In this case ENTER↑ is used to separate the input variables x and y. To input both x and y you would key in x, press ENTER↑, key in y and press A.
[x] A	The box around the variable x indicates input by pressing STO A.
(x) A	Parentheses indicate an option. In this case, x is not a required input but could be input in special cases.
➜x A	➜ is the symbol for calculate. This indicates that you may calculate x by pressing key A.
➜x, y, z A	This indicates that x, y, and z are calculated by pressing A once. The values would be printed in x, y, z order.
➜x; y; z A	The semi-colons indicate that after x has been calculated using A, y and z may be calculated by pressing R/S.
➜"x", y A	The quote marks indicate that the x value will be "paused" or held in the display for one second. The pause will be followed by the display of y.
◁▷ x A	The two-way arrow ◁▷ indicates that x may be either output or input when the associated user-definable key is pressed. If numeric keys have been pressed between user-definable keys, x is stored. If numeric keys have not been pressed, the program will calculate x.

SYMBOLS AND CONVENTIONS (Continued)

SYMBOL OR CONVENTION	INDICATED MEANING
P? [A]	The question mark indicates that this is a mode setting, while the mnemonic indicates the type of mode being set. In this case a print mode is controlled. Mode settings typically have a 1.00 or 0.00 indicator displayed after they are executed. If 1.00 is displayed, the mode is on. If 0.00 is displayed, it is off.
START [A]	The word START is an example of a command. The start function should be performed to begin or start a program. It is included when initialization is necessary.
DEL [A]	This special command indicates that the last value or set of values input may be deleted by pressing [A].
➜x; ... [A]	Three dots (...) indicate that additional output follows. See User Instructions for complete description of variables output.

HEWLETT hp PACKARD

1000 N.E. Circle Blvd., Corvallis, OR 97330

A B C ● E Rev. 3/78