GENERAL FINANCE STATISTICS

PROGRAM LIBRARY

sinclair

Enterprise Programmable

QUICK GUIDE TO USING THE LIBRARY

Always refer to the instruction book until you are familiar with the use of the calculator.

To enter a program:

> Enter goto/0/0/prog
>
> Then enter the keystrokes as given in the table on the right hand side of each program in the library.
>
> Then enter prog/goto/0/1

Always remember to press ▲ when the upper case of a key is required.

To use a program follow the pre-execution (if applicable) and execution sequence given with it. Remember to wait till the display lights up before entering a number in the middle of an execution sequence.

If you think you have made a mistake in program entry, check the program with some data for which you know the correct answer. If there is an error, either re-enter the program, or find the error using the check codes and correct it as detailed in the instruction book.

If you make a mistake in the execution sequence, it is generally necessary to enter C/goto/0/1 and to start the pre-execution and execution sequences again.

It is a good idea to press C to clear any previous results before starting an execution sequence or, indeed, any calculation.

A program can be halted in the middle of execution by entering /stop/ (i.e. pressing ▲ +/–).

CONTENTS

Section

GENERAL

Hints 1
Conversions 2
Time 3
Games 4

FINANCE

Discounts Markup & Tax 5
Compound interest - single sums 6
Compound interest - regular payments 7
Interest rates 8
Mortgages 9
Business methods 10

STATISTICS

Sampling 11
Significance Tests 12
Quality Control 13
Distributions 14
Transformations 15
Random numbers 16

1. HINTS

This is a mixture of programs and hints to help with calculator use and programming technique

1.1 Holding extra constants in memory
1.2 % Functions
1.3 Absolute value
1.4 Integer part
1.5 Memory functions
1.6 Memory exchange
1.7 Brackets
1.8 Roots
1.9 Powers of negative numbers

HOLDING EXTRA CONSTANTS IN MEMORY

Store constants in program memory as opposite.
For instance, 41.37525 is stored starting at 50. So to recall 41.37525 press goto/5/0/run. The effect is the same as using rcl n when the constant is stored in memory n. To recall the same constant for a second time it is only necessary to press /run/.

To save space in the program the goto statements opposite can be omitted, in which case it is always necessary to press /goto/n/m/run/ to recall the constant stored starting at nm

KEY	#	KEY	#
HALT	00	+/−	40
2	01	stop	41
./EE	02	goto	42
9	03	3	43
9	04	0	44
7	05		45
9	06		46
2	07		47
5	08		48
./EE	09		49
8	10	4	50
goto	11	1	51
0	12	./EE	52
0	13	3	53
	14	7	54
1	15	5	55
./EE	16	2	56
0	17	5	57
7	18	stop	58
4	19	goto	59
8	20	5	60
3	21	0	61
stop	22		62
goto	23		63
1	24		64
5	25		65
	26		66
	27		67
	28		68
	29		69
7	30		70
./EE	31		71
9	32		72
6	33		73
6	34		74
9	35		75
5	36		76
./EE	37		77
1	38		78
1	39		79

PERCENTAGE FUNCTIONS

KEY	#	KEY	#
HALT	00		40
(	01		41
×	02		42
stop	03		43
÷	04		44
1	05		45
0	06		46
0	07		47
)	08		48
goto	09		49
0	10		50
0	11		51
	12		52
	13		53
	14		54
	15		55
	16		56
	17		57
	18		58
	19		59
	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

Execution:

x/run/y/run/y% of x
x/+/run/y/run/y% of
x/=/x + y% of x
x/−/run/y/run/y% of
x/=/x − y% of x

ABSOLUTE VALUE

The absolute value, |x|, of a number, x is the positive value of x, so

|15.87| = 15.87 and
|−84.5| = 84.5

This can be found by first squaring then taking the square root. There is no need for this as a program by itself, but it can be useful within a program.

Execution:

x/run/|x|

KEY	#	KEY	#
HALT	00		40
x^2	01		41
$\sqrt{x}$	02		42
goto	03		43
0	04		44
0	05		45
	06		46
	07		47
	08		48
	09		49
	10		50
	11		51
	12		52
	13		53
	14		54
	15		55
	16		56
	17		57
	18		58
	19		59
	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

INTEGER PART

KEY	#	KEY	#
HALT	00		40
+	01		41
1	02		42
./EE	03		43
./EE	04		44
9	05		45
—	06		46
1	07		47
./EE	08		48
./EE	09		49
9	10		50
=	11		51
goto	12		52
0	13		53
0	14		54
	15		55
	16		56
	17		57
	18		58
	19		59
	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

The integer part of 15.586 is 15, of 121.137 is 121.

It is sometimes necessary to find this within a program – this can be done by adding and then subtracting a very large number.

Obviously there is no need for this as a program by itself.

Execution:

x/run/integer part of x

MEMORY FUNCTIONS

M + n:
There is a key for this anyway — it adds the displayed number to memory n

M — n:
Subtract displayed number from the number in memory, keep the result in the memory and leave the display unaltered.

M × n:
Multiply the display number by the number in memory, keep the result in memory and leave the display unaltered.

M ÷ n:
Divide the number in memory by the displayed number, store the result and leave the display unaltered.

Execution:

M — : x/—/run/x
M × : x/×/run/x
M ÷ : x/÷/run/x

KEY	#	KEY	#
HALT	00		40
x ◂▸ y	01		41
rcl	02		42
n	03		43
x ◂▸ y	04		44
=	05		45
sto	06		46
n	07		47
x ◂▸ y	08		48
goto	09		49
0	10		50
0	11		51
	12		52
	13		53
	14		54
	15		55
	16		56
	17		57
	18		58
	19		59
	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

MEMORY EXCHANGE

KEY	#	KEY	#
HALT	00		40
x◂▸y	01		41
rcl	02		42
n	03		43
x◂▸y	04		44
sto	05		45
n	06		46
x◂▸y	07		47
goto	08		48
0	09		49
0	10		50
	11		51
	12		52
	13		53
	14		54
	15		55
	16		56
	17		57
	18		58
	19		59
	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

To exchange the contents of the display with the contents of a memory without using another memory.

This might be used as part of a program, or in calculate mode.

Execution:

x/run/y

where y was in memory n

BRACKETS

After an opening bracket the display still contains the number it did before the opening bracket. Suppose x is in memory 0 then

$\frac{1}{x+1}$ could be programmed as in

00–13 opposite, or as 20–31 opposite. This useful feature saves program steps.

Note:

It could also be programmed as in 40–48 opposite, saving even more steps. The formula has been re-written as

$$\frac{1}{1+\frac{1}{x}}$$

and so will give an error if x = 0.

KEY	#	KEY	#
HALT	00	1/x	40
rcl	01	+	41
0	02	1	42
÷	03	=	43
(	04	1/x	44
rcl	05	stop	45
0	06	goto	46
+	07	4	47
1	08	0	48
)	09		49
=	10		50
goto	11		51
0	12		52
0	13		53
	14		54
	15		55
	16		56
	17		57
	18		58
	19		59
rcl	20		60
0	21		61
÷	22		62
(	23		63
+	24		64
1	25		65
)	26		66
=	27		67
stop	28		68
goto	29		69
2	30		70
0	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

ROOTS

KEY	#	KEY	#
HALT	00		40
y^x	01		41
stop	02		42
1/x	03		43
=	04		44
goto	05		45
0	06		46
0	07		47
	08		48
	09		49
	10		50
	11		51
	12		52
	13		53
	14		54
	15		55
	16		56
	17		57
	18		58
	19		59
	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

$\sqrt[x]{y}$ is $y^{1/x}$

Execution:

y/run/x/run/$\sqrt[x]{y}$

POWERS OF NEGATIVE NUMBERS

The y^x key will not find powers of negative numbers. This program finds integer powers of negative numbers (for instance $(-3)^3 = -27$).

For fractional powers of negative numbers see the section on complex numbers.

Execution:

y/run/x/run/y^x

KEY	#	KEY	#
HALT	00	rcl	40
gin	01	1	41
1	02	=	42
0	03	goto	43
y^x	04	0	44
stop	05	0	45
=	06	rcl	46
goto	07	0	47
0	08	y^x	48
0	09	rcl	49
x^2	10	1	50
$\sqrt{x}$	11	=	51
sto	12	+/−	52
0	13	goto	53
stop	14	0	54
sto	15	0	55
1	16		56
÷	17		57
2	18		58
−	19		59
(	20		60
+	21		61
1	22		62
./EE	23		63
./EE	24		64
9	25		65
−	26		66
1	27		67
./EE	28		68
./EE	29		69
9	30		70
)	31		71
=	32		72
+/−	33		73
gin	34		74
4	35		75
6	36		76
rcl	37		77
0	38		78
y^x	39		79

2. CONVERSIONS

To perform just one or two conversions the best thing to do is extract the conversion factor from the programs and multiply or divide by it as appropriate. The programs are useful when a lot of conversions need doing.

If necessary the various program segments can be re-arranged to give the combination of conversions one is interested in using.

2.1 Feet and inches to metres and vice versa

2.2 Imperial to metric
Feet and inches to metres
Pounds and ounces to kilograms
gallons to litres
pints to litres
fluid ounces to centilitres

2.3 Metres to feet and inches
Kilograms to pounds and ounces
Litres to gallons

2.4 Degrees Fahrenheit to degrees centigrade and vice versa
Inches of mercury to millibars and vice versa
Knots to metres per second and vice versa

2.5 Miles to kilometres and vice versa
Miles per hour to kilometres per hour and vice versa
Miles per gallon to litres per 100 kilometres and vice versa
Gallons to litres and vice versa
Pints to litres and vice versa
Pounds per square inch to kilograms per square metre and vice versa
Pounds per square inch to Bar and vice versa

2.6 Miles to kilometres and vice versa
Acres to hectares and vice versa
Chains to poles to metres
Metres to poles
Degrees minutes and seconds to decimal degrees.

For most of the conversions, to convert a second or subsequent quantity, one need only enter the quantity and press /=/. Such conversions have been marked *.

FEET AND INCHES TO METRES AND VICE VERSA

Execution:

goto/0/1/ft/run/ins/run/m
goto/1/9/m/run/ft/run/ins

To do another conversion the same type it is not necessary to start with the goto.

KEY	#	KEY	#
HALT	00	)	40
+	01	×	41
(	02	1	42
stop	03	2	43
×	04	=	44
1	05	stop	45
2	06	goto	46
)	07	1	47
×	08	9	48
./EE	09		49
0	10		50
2	11		51
5	12		52
4	13		53
=	14		54
stop	15		55
goto	16		56
0	17		57
1	18		58
×	19		59
3	20		60
./EE	21		61
2	22		62
8	23		63
0	24		64
8	25		65
—	26		66
(	27		67
+	28		68
1	29		69
./EE	30		70
./EE	31		71
9	32		72
—	33		73
1	34		74
./EE	35		75
./EE	36		76
9	37		77
=	38		78
stop	39		79

IMPERIAL TO METRIC

2.2

KEY	#	KEY	#
HALT	00	0	40
×	01	=	41
1	02	stop	42
2	03	goto	43
+	04	3	44
stop	05	4	45
×	06	×	46
./EE	07	./EE	47
0	08	5	48
2	09	6	49
5	10	8	50
4	11	2	51
=	12	5	52
stop	13	=	53
goto	14	stop	54
0	15	goto	55
1	16	4	56
×	17	6	57
1	18	×	58
6	19	2	59
+	20	./EE	60
stop	21	8	61
×	22	4	62
./EE	23	1	63
0	24	3	64
2	25	=	65
8	26	stop	66
3	27	goto	67
5	28	5	68
=	29	8	69
stop	30		70
goto	31		71
1	32		72
7	33		73
×	34		74
4	35		75
./EE	36		76
5	37		77
4	38		78
6	39		79

Execution:

goto/0/1/ft/run/ins/run/metres
goto/1/7/lb/run/oz/run/kg
•goto/3/4/gallons/run/litres
•goto/4/6/pints/run/litres
•goto/5/8/fl oz/run/centilitres

METRIC TO IMPERIAL

KEY	#	KEY	#
HALT	00	+	40
×	01	1	41
3	02	./EE	42
./EE	03	./EE	43
2	04	9	44
8	05	—	45
0	06	1	46
8	07	./EE	47
—	08	./EE	48
(	09	9	49
+	10	=	50
1	11	stop	51
./EE	12	)	52
./EE	13	×	53
9	14	1	54
—	15	6	55
1	16	=	56
./EE	17	stop	57
./EE	18	goto	58
9	19	3	59
=	20	1	60
stop	21	×	61
)	22	./EE	62
×	23	2	63
1	24	1	64
2	25	9	65
=	26	9	66
stop	27	8	67
goto	28	=	68
0	29	stop	69
1	30	goto	70
×	31	6	71
2	32	1	72
./EE	33		73
2	34		74
0	35		75
4	36		76
6	37		77
—	38		78
(	39		79

Execution:

goto/0/1/metres/run/ft/run/ins
goto/3/1/kg/run/lb/run/oz
*goto/6/1/litres/run/gallons

To perform another conversion of the same sort it is not necessary to start with the goto.

METEOROLOGY

KEY	#	KEY	#
HALT	00	./EE	40
—	01	8	41
3	02	6	42
2	03	4	43
÷	04	=	44
1	05	stop	45
./EE	06	goto	46
8	07	3	47
=	08	7	48
stop	09	×	49
goto	10	./EE	50
0	11	5	51
1	12	1	52
×	13	4	53
1	14	4	54
./EE	15	4	55
8	16	=	56
+	17	stop	57
3	18	goto	58
2	19	4	59
=	20	9	60
stop	21	÷	61
goto	22	./EE	62
1	23	5	63
3	24	1	64
×	25	4	65
3	26	4	66
3	27	4	67
./EE	28	=	68
8	29	stop	69
6	30	goto	70
4	31	6	71
=	32	1	72
stop	33		73
goto	34		74
2	35		75
5	36		76
÷	37		77
3	38		78
3	39		79

Execution:

goto/0/1/°F/run/°C
goto/1/3/°C/run/°F
•goto/2/5/ins Hg/run/mB
•goto/3/7/mB/run/ins Hg
•goto/4/9/knots/run/ms^{-1}
•goto/6/1/ms^{-1}/run/knots

To convert a new quantity of the same unit it is not necessary to start with the /goto/.

Note:

10 mB = 1 kPa

MOTORING CONVERSIONS

Execution:

*goto/0/1/miles/×/run/km

*goto/0/1/mph/×/run/km hr^{-1}

*goto/0/1/km/÷/run/miles

*goto/0/1/km hr^{-1}/÷/run/mph

goto/1/2/mpg/run/litres per 100 km

goto/1/2/litres per 100 km/run/mpg

goto/2/5/gallons/×/run/litres

goto/2/5/litres/÷/run/gallons

*goto/3/5/pints/×/run/litres

*goto/3/5/litres/÷/run/pints

*goto/4/6/lb sq in^{-1}/×/run/kg m^{-2}

*goto/4/6/kg m^{-2}/÷/run/lb sq in^{-1}

*goto/5/7/lb sq in^{-1}/×/run/Bars

*goto/5/7/Bars/÷/run/lb sq in^{-1}

It is not necessary to start with the /goto/ when doing another conversion of the same type.

KEY	#	KEY	#
HALT	00	5	40
1	01	=	41
./EE	02	stop	42
6	03	goto	43
0	04	3	44
9	05	5	45
3	06	7	46
=	07	0	47
stop	08	3	48
goto	09	./EE	49
0	10	0	50
1	11	7	51
—	12	=	52
2	13	stop	53
8	14	goto	54
2	15	4	55
./EE	16	6	56
4	17	./EE	57
8	18	0	58
=	19	6	59
1/x	20	8	60
stop	21	9	61
goto	22	4	62
1	23	8	63
2	24	=	64
4	25	stop	65
./EE	26	goto	66
5	27	5	67
4	28	7	68
6	29		69
=	30		70
stop	31		71
goto	32		72
2	33		73
5	34		74
./EE	35		75
5	36		76
6	37		77
8	38		78
2	39		79

SURVEYING ETC.

KEY	#	KEY	#
HALT	00	./EE	40
1	01	1	41
./EE	02	9	42
6	03	8	43
0	04	8	44
9	05	4	45
3	06	=	46
=	07	stop	47
stop	08	goto	48
goto	09	3	49
0	10	9	50
1	11	×	51
./EE	12	6	52
4	13	0	53
0	14	+	54
4	15	stop	55
6	16	×	56
9	17	6	57
=	18	0	58
stop	19	+	59
goto	20	stop	60
1	21	÷	61
2	22	3	62
×	23	6	63
4	24	0	64
+	25	0	65
stop	26	=	66
×	27	stop	67
5	28	goto	68
./EE	29	5	69
0	30	1	70
2	31		71
9	32		72
2	33		73
=	34		74
stop	35		75
goto	36		76
2	37		77
3	38		78
×	39		79

Execution:

·goto/0/1/miles/×/run/km

·goto/0/1/km/÷/run/miles

·goto/1/2/acres/×/run/hectares

·goto/1/2/hectares/÷/run/acres

goto/2/3/chains/run/poles/
run/metres

·goto/3/9/metres/run/poles

goto/5/1/degrees/run/
minutes/run/
seconds/run/decimal degrees

To do another conversion of the same type one does not need to start with the /goto/.

3. TIME

3.1 Universal Calendar

3.2 Date of Easter

3.3 Timer

3.4 Stopwatch

UNIVERSAL CALENDAR

This program finds the day of the week given the date. It is set for dates from 1st March 1900 to 28th February 2100. For dates from 1st March 2100 to 28th February 2200 substitute 2943 for steps 56–59 in the program. For Western European dates from 1st March 1800 to 28th February 1900 substitute 2591, for 14th September 1752 to 28th February 1800 substitute 2471. For dates before 1752 in England (and for some dates after that in other countries) historical methods must be used to find the day of the week, because of the variations in calendars and the date of New Year's day.

Execution:

day/run/month/run/year in full/run/day of week

Jan = 1, Feb = 2 etc.

in answer Sun = 0, Mon = 1, Tue = 2, . . . , Sat = 6.

Example:

26/run/12/run/1976/0

So 26th December 1976 was a Sunday.

KEY	#	KEY	#
HALT	00	—	40
sto	01	1	41
0	02	./EE	42
stop	03	./EE	43
sto	04	9	44
1	05	+	45
stop	06	(	46
sto	07	rcl	47
2	08	2	48
rcl	09	×	49
1	10	1	50
—	11	./EE	51
3	12	2	52
=	13	5	53
gin	14	)	54
2	15	—	55
2	16	2	56
+	17	7	57
4	18	1	58
goto	19	1	59
3	20	+	60
1	21	rcl	61
1	22	0	62
+/—	23	+	63
M+	24	7	64
2	25	=	65
rcl	26	gin	66
1	27	6	67
+	28	5	68
1	29	+	69
3	30	1	70
×	31	./EE	71
2	32	./EE	72
./EE	33	9	73
6	34	—	74
+	35	1	75
1	36	./EE	76
./EE	37	./EE	77
./EE	38	9	78
9	39	=	79

DATE OF EASTER DAY 1900-2099

Table to find Easter 1900–2099

Golden number	Day and month		Sun letter
—	March	21	C
14		22	D
3		23	E
—		24	F
11		25	G
—		26	A
19		27	B
8		28	C
—		29	D
16		30	E
5		31	F
—	April	1	G
13		2	A
2		3	B
—		4	C
10		5	D
—		6	E
18		7	F
7		8	G
—		9	A
15		10	B
4		11	C
—		12	D
12		13	E
1		14	F
—		15	G
9		16	A
17		17	B
6		18	C
—		19	D
—		20	E
—		21	F
—		22	G
—		23	A
—		24	B
—		25	C

This program finds the Golden number and the Sunday letter for a given year; the date of Easter can then be obtained from the attached table.

Use the program to find the Sunday letter and the Golden Number.

Locate the Golden Number in the first column of the Table and read across to find the date of the Pashal Full Moon in the second column.

Read down the third column from the day following the Paschal Full Moon to find the Sunday letter. The date opposite this letter in column 2 is the date of Easter Sunday.

e.g. 1976 Golden number = 1
Sunday letter = C

Column 1 gives Pashal Full Moon as April 14. First C below April 14 is April 18.
Therefore April 18 = Easter Sunday.

Note:

The Dominical letter determines the date of the first Sunday after January 1st. (The dominical letter is A if January 1st is a Sunday, B if January 2nd is a Sunday etc.)

Thus the Dominical letter is the same as the Sunday letter except in a leap year. In a leap year the Dominical letter is the one after the Sunday letter (so if the Sunday letter is D then the Dominical letter is E and January 5th is a Sunday).

Execution:

year in full/run/golden number/run/Sunday letter (as a number)

Number	Sunday letter
1	A
2	B
3	C
4	D
5	E
6	F
7	G

Example:

1901/run/2/run/6

So the Paschal Full Moon in 1901 is on April 3rd, the Sunday letter is F and Easter Day is April 7th.

KEY	#	KEY	#
HALT	00	−	40
sto	01	1	41
0	02	9	42
−	03	=	43
2	04	gin	44
1	05	5	45
0	06	0	46
7	07	goto	47
÷	08	4	48
/EE	09	0	49
8	10	+	50
+	11	2	51
7	12	0	52
+	13	=	53
gin	14	stop	54
1	15	rcl	55
2	16	1	56
(	17	stop	57
+/−	18	goto	58
+	19	0	59
1	20	1	60
=	21		61
gin	22		62
1	23		63
9	24		64
)	25		65
=	26		66
+/−	27		67
+	28		68
8	29		69
=	30		70
sto	31		71
1	32		72
rcl	33		73
0	34		74
−	35		75
1	36		76
9	37		77
0	38		78
0	39		79

TIMER

KEY	#	KEY	#
HALT	00	6	40
1	01	=	41
./EE	02	sto	42
1	03	0	43
1	04	stop	44
1	05	+/−	45
1	06	sto	46
1	07	1	47
1	08	stop	48
1	09	rcl	49
sto	10	1	50
0	11	+	51
9	12	rcl	52
9	13	0	53
9	14	=	54
9	15	sto	55
+/−	16	1	56
sto	17	gin	57
1	18	4	58
goto	19	9	59
4	20	goto	60
8	21	4	61
9	22	4	62
9	23		63
9	24		64
9	25		65
+	26		66
rcl	27		67
1	28		68
=	29		69
1/x	30		70
×	31		71
6	32		72
6	33		73
6	34		74
./EE	35		75
6	36		76
6	37		77
6	38		78
6	39		79

Enter an amount of time in seconds and the display lights up that amount of time later.

Execution:

(a) To calibrate the timer (essential before using it) run/run/, precisely 10 minutes after this: stop/goto/2/2/run.

(b) To use: time in seconds, t/run/run/ display lights up t seconds later.

Warning: this timer is not very accurate, especially if the calculator is being run off batteries.

After first use, timer may be re-used without re-calibration – just repeat execution sequence (b). To re-calibrate: goto/0/1/ then execution sequence (a)

STOPWATCH

Finds the time elapsed in seconds.

Execution:

(a) To calibrate (essential before use) run/run/wait exactly 10 minutes then:
stop/goto/1/8/run/

(b) To use:
run/stop/goto/3/9/run/time in seconds between pressing run and stop

Warning: the stopwatch is not very accurate, especially if the calculator is being run off batteries.

After first use stopwatch may be used again without recalibration – just repeat execution (b). To recalibrate goto/0/1 then execution sequence (a).

KEY	#	KEY	#
HALT	00	1	40
1	01	x ◂▸ y	41
./EE	02	0	42
1	03	sto	43
1	04	1	44
1	05	x ◂▸ y	45
1	06	stop	46
1	07	rcl	47
1	08	0	48
1	09	M+	49
sto	10	1	50
0	11	goto	51
0	12	4	52
sto	13	7	53
1	14		54
goto	15		55
4	16		56
6	17		57
6	18		58
6	19		59
6	20		60
./EE	21		61
6	22		62
6	23		63
6	24		64
6	25		65
6	26		66
÷	27		67
rcl	28		68
1	29		69
=	30		70
sto	31		71
0	32		72
0	33		73
sto	34		74
1	35		75
goto	36		76
4	37		77
6	38		78
rcl	39		79

4. GAMES

4.1 Dice
4.2 Matchsticks
4.3 Sequences
4.4 Guess the number
4.5 Moonlanding
4.6 Torpedoes
4.7 Sailing a supertanker

PSEUDO-RANDOM DICE THROWER

This dice is slightly biased, but not too heavily to be convincing!

Execution:

Choose any starting value x between 0 and 1.
x/run/d_1/run/d_2/run/d_3/etc.

where d_1, d_2, d_3 are successive 'throws'.

KEY	#	KEY	#
HALT	00		40
×	01		41
1	02		42
0	03		43
1	04		44
÷	05		45
1	06		46
7	07		47
+	08		48
(	09		49
+/−	10		50
+	11		51
1	12		52
=	13		53
gin	14		54
1	15		55
3	16		56
sto	17		57
0	18		58
)	19		59
=	20		60
stop	21		61
rcl	22		62
0	23		63
goto	24		64
0	25		65
1	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

MATCH-STICK GAME

KEY	#	KEY	#
HALT	00		40
−	01		41
(	02		42
+/−	03		43
+	04		44
1	05		45
+	06		46
4	07		47
=	08		48
gin	09		49
0	10		50
8	11		51
+/−	12		52
gin	13		53
1	14		54
8	15		55
−	16		56
3	17		57
+	18		58
4	19		59
)	20		60
−	21		61
stop	22		62
=	23		63
stop	24		64
goto	25		65
0	26		66
1	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

You put N matchsticks down on the table. At each turn, each player may pick up 1, 2 or 3 matchsticks; because you choose the starting number N, the machine has the first turn. The object of the game is to avoid picking up the last matchstick; thus if either player leaves 1 matchstick after his turn he has won.

Execution:

N/run/machine plays
/1, 2 or 3/run/you play
/run/machine plays
/1, 2 or 3/run/you play
etc.

Display each time shows number of matchsticks remaining.

Example:

85/run/84/2/run/
82/run/81 . . . etc.

SEQUENCE GAME

Enter first five numbers in sequence.
Machine will try to guess the sixth.

Execution:

1st number/run/2nd number/run/3rd number/run/4th number/run/5th number/run/machine's guess

Example:

0/run/2/run/4/run/6/run/8/run/10

KEY	#	KEY	#
HALT	00		40
×	01		41
2	02		42
+/−	03		43
+	04		44
(	05		45
stop	06		46
×	07		47
5	08		48
)	09		49
+	10		50
(	11		51
stop	12		52
×	13		53
2	14		54
+/−	15		55
)	16		56
+	17		57
(	18		58
stop	19		59
×	20		60
4	21		61
+/−	22		62
)	23		63
+	24		64
(	25		65
stop	26		66
×	27		67
4	28		68
)	29		69
=	30		70
goto	31		71
0	32		72
0	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

GUESS THE NUMBER GAME

KEY	#	KEY	#
HALT	00	)	40
sto	01	=	41
1	02	sto	42
—	03	3	43
rcl	04	rcl	44
2	05	1	45
sto	06	goto	46
0	07	0	47
=	08	1	48
x^2	09	+/—	49
+/—	10	sto	50
gin	11	1	51
6	12	goto	52
1	13	6	53
stop	14	4	54
rcl	15	+/—	55
6	16	sto	56
stop	17	0	57
sto	18	goto	58
1	19	6	59
rcl	20	4	60
3	21	1	61
×	22	M+	62
9	23	6	63
9	24	rcl	64
—	25	0	65
(	26	—	66
+	27	rcl	67
1	28	1	68
./EE	29	=	69
./EE	30	gin	70
9	31	4	71
—	32	9	72
1	33	+/—	73
./EE	34	gin	74
./EE	35	5	75
9	36	5	76
=	37	0	77
sto	38	rcl	78
2	39	0	79

Gives the highest common factor of your guess and a random number between 1 and 100.

Pre-execution:

To start store any number between 0 and 1 in M3. Enter first guess and goto 18 (clear memory 6, which contains total number of guesses).

Execution:

run/hcf enter new guess/run/. . . etc.

If you have guessed right display will show 0.

To try a new number

goto/1/8/guess/run/. . . etc.

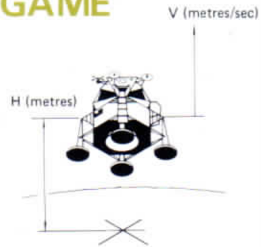
To reduce your cumulative score to zero, store 0 in memory 6.

Example:

.123456/sto/3/goto/1/8/15/run/3/12/run/0

3 was the hcf of 12 and 15.
12 was the right guess.

MOON-LANDING GAME



The object of the game is to land the lunar module on the moon starting at a given height and a given (upwards) velocity.

The payload of the lunar module is 1000 kg and the power of the engine (thrust ratio) is 2400 Newtons per kg of fuel.

After entering the program, start by entering the initial values.

Height (in metres)	/sto/5
Velocity (upwards, in ms^{-1})	/sto/3
Mass (payload plus fuel mass in kg)	/sto/1/sto/2
Thrust ratio (in Nkg^{-1})	/sto/6

Suggested starting values are

Height	15 000 m
Velocity	$-200\ ms^{-1}$
(ie 200 ms^{-1}downwards)	
Mass	1600 kg
(i.e. a fuel mass of 600 kg)	
Thrust ratio	2400 N kg^{-1}

KEY	#	KEY	#
HALT	00	5	40
sto	01	rcl	41
0	02	0	42
×	03	sto	43
2	04	0	44
+/−	05	×	45
+	06	1	46
rcl	07	./EE	47
1	08	6	48
=	09	2	49
sto	10	+/−	50
2	11	=	51
1/x	12	M+	52
×	13	3	53
rcl	14	÷	54
1	15	2	55
=	16	+/−	56
ln	17	+	57
×	18	rcl	58
rcl	19	3	59
6	20	stop	60
=	21	×	61
M+	22	rcl	62
3	23	0	63
×	24	=	64
rcl	25	M+	65
2	26	5	66
+/−	27	rcl	67
÷	28	5	68
2	29	stop	69
+	30	rcl	70
(	31	2	71
rcl	32	sto	72
0	33	1	73
×	34	−	74
rcl	35	1	75
6	36	0	76
)	37	0	77
=	38	0	78
M+	39	=	79

Landing is a succession of "burns" (when the engine is on and consumes fuel at a rate of 2 kgs^{-1}) and "coasts" (when the engine is off and the LEM falls freely). One has to reach a height of 0 and a velocity of zero before the fuel runs out. The landing tolerance is ±.5 ms^{-1} for velocity and ±.5 m for height. If you land outside this tolerance, you have crashed. Landing must be achieved before the fuel runs out.

When you have tried the game, you might like to change the starting values – to make the game more difficult – increase the height and velocity, decrease the amount of fuel available and the thrust ratio, to make it easier do the opposite.

The equations used are exact.

m = mass

F = mass of fuel available

M_p = payload mass = 1000 kg

$m = M_p + F$

f = rate of fuel consumption in burn

$= 2\ \mathrm{kgs}^{-1}$

T = thrust of engines = 4800 N

$\frac{T}{f}$ = thrust ratio

H = height

V = velocity

t = length of burn or coast in seconds

ℓ = lunar gravity = 1.62 $\mathrm{N\,kg}^{-1}$

For Coast

new mass $= m' = m$

new height $= H + Vt - \frac{1}{2}\ell t^2$

new velocity $= V - \ell t$

For burn

new mass $= m' = m - ft$

new height $= H + Vt - \frac{1}{2}\ell t^2 + \frac{T}{f}t - \frac{T}{f^2}\ m' \ln \frac{m}{m'}$

new velocity $= V - \ell t + \frac{T}{f} \ln \frac{m}{m'}$

Execution:

For burns:

goto/0/1/length of burn in seconds/run/velocity, V/run/
/height, H/run/fuel left (in kg), F/

For coasts:

goto/4/3/length of coast in seconds/run/V/run/H/run/F

Example of the start of a game

mass 1600/sto/1/sto/2
velocity 190/ +/−/sto/3
height 15000/sto/5
thrust ratio 2400/sto/6

goto/4/3/3/run/− 194.8/run/14422.7/run/600/goto/0/1/
10/run/− 180.8/run/12845.6/run/580/ . . .

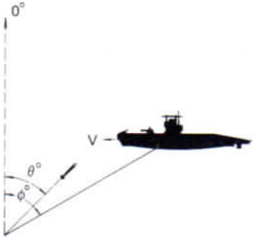
The results tabulated:

Time		Velocity	Height	Fuel left
3	coast	− 194.8	14422.7	600
10	burn	− 180.8	12845.6	580
1	coast	− 182.5	12664.0	580

etc.

SUBMARINE GAME

KEY	#	KEY	#
HALT	00	−	40
rcl	01	rcl	41
2	02	0	42
×	03	=	43
9	04	sin	44
./EE	05	+	45
7	06	(	46
−	07	rcl	47
(	08	1	48
+	09	×	49
1	10	rcl	50
./EE	11	0	51
./EE	12	cos	52
9	13	÷	53
−	14	5	54
1	15	0	55
./EE	16	)	56
./EE	17	=	57
9	18	x^2	58
=	19	−	59
sto	20	0	60
1	21	./EE	61
)	22	0	62
=	23	0	63
sto	24	1	64
2	25	=	65
×	26	gin	66
1	27	7	67
8	28	2	68
0	29	goto	69
−	30	0	70
9	31	1	71
0	32	8	72
=	33	8	73
sto	34	8	74
0	35	8	75
stop	36	8	76
rcl	37	8	77
1	38	8	78
stop	39	8	79



Your sonar spots an enemy submarine heading across your course at a bearing ϕ° and at a speed of v knots. You fire a torpedo at a bearing θ°.

Pre-execution:

Store any number between 0 and 1 in memory 2.

Execution:

run/ϕ/run/v/

choose your θ:θ/run/

If you hit: 88888888 is displayed. Now you can try for another submarine:

run/ϕ/run/v/and continue as before.

If you miss, the new ϕ is displayed immediately:

ϕ/run/v/

Now choose another θ and try again as before.

The torpedo homes on the submarine and will hit it if fired at the right bearing, plus or minus a few degrees (it is also always more likely to hit if the submarine is near broadside to you). The speed of the torpedo is 50 knots (it's jet propelled!). You may want to make the game more difficult – by making the torpedo slower (change steps 54 and 55 of the program to say, 30 knots), or less sophisticated (change steps 60 to 64 to, say, .0002, for a torpedo which homes less well and must be fired on a more accurate bearing).

Your ship is more or less stationary.

Example:

```
0.1 2 3 4 5 6 7 /sto/2
run/φ = −54.44°/
run/v = 1 knot
try θ  = −54/run/8888888
          – a hit!
run (again) φ = 74.88°/
run/v = 1 knot/
try θ  = 73/run/88888888
          – another hit.
run/φ = 69.4°/run/v = 8 knots/
try θ = 59°/run/16.253586 – a miss:
your new φ is 16.25°
run/v = 8 knots/try another θ.
```

SAILING A SUPERTANKER ACROSS THE ATLANTIC

KEY	#	KEY	#
HALT	00	2	40
0	01	0	41
sto	02	)	42
6	03	×	43
sto	04	4	44
3	05	+/−	45
sto	06	+	46
2	07	(	47
stop	08	rcl	48
sto	09	1	49
1	10	×	50
stop	11	rcl	51
sto	12	0	52
0	13	×	53
÷	14	4	54
4	15	)	55
+/−	16	=	56
=	17	M+	57
e^x	18	3	58
+/−	19	rcl	59
+	20	0	60
1	21	M+	61
×	22	6	62
(	23	rcl	63
rcl	24	3	64
1	25	goto	65
×	26	0	66
4	27	8	67
−	28		68
rcl	29		69
2	30		70
)	31		71
+	32		72
(	33		73
+	34		74
rcl	35		75
2	36		76
=	37		77
stop	38		78
sto	39		79

You may set the engine speed E to any value from −8 (full speed astern to +8 (full speed ahead). For instance ½ speed ahead is +4, dead slow astern is −1. (1 +/−). You may keep this speed for .001 hrs (about 4 secs) to 300 hrs (the fuel capacity of the ship).

You have to cross the Atlantic (2000 nautical miles) and come to a stop (less than .01 knots) within 1/10 Nm of the mooring buoy. To see how long you took – rcl/6.

Expert score 68 hours

Very good 77 hours

Execution:

goto/0/1/run/engine speed/run/time/run/ship's speed in knots/run/distance travelled in nautical miles/new engine speed/run/time/run/ship's speed/run/distance/new engine speed. etc.

Example:

4.7 (ii)

Full speed ahead for 64 hours and then some manoeuvering:

goto/0/1/run/8/run/64/run/31.999/run/1920 – your speed is almost 32 knots and you have travelled 1920 nautical miles – so full speed astern for 2 hours.
8/+/–/run/2/run/6.81799/run/1956.72

Try half speed ahead for four hours
4/run/4/run/12.62/run/1997

Almost there now, but travelling a bit fast. How long will it take to get the speed down and arrive within 0.1 nautical mile of the mooring bouy?

5. FINANCE DISCOUNTS MARK-UP AND TAX

The astute programmer will realise that discount, mark-up and flat rate tax are the same thing and will be able to use the following nine programs interchangeably.

5.1 Mark-up (x + a% of x)

5.2 Discount (x − a% of x)

5.3 Mark-up from gross %
(y = x + a% of y)

5.4 Discount from net %
(y = x − a% of y)

5.5 Percentage change arising from a change in mark-up or discount

5.6 Flat rate tax – finds mark-ups and grand totals – fixed tax rates

5.7 Flat rate tax – as 6., variable rates

5.8 Flat rate tax – finds mark-up given marked-up price and finds grand totals

5.9 As for 8, but with variable tax rate

MARK-UP

Execution:

% mark-up/run/price/run/marked up price/
price/run/marked-up price

To enter new mark up:

goto/0/1/new mark-up/run/. . . etc

Example:

I wish to mark-up my goods by 30%. Their costs were £1.53, £5.46 and £6.78 respectively

30/run/1.53/run/1.989/5.46/run/7.098/6.78/run/8.814

So the marked-up prices are £1.99, £7.10, £8.81 respectively.

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
+	05		45
1	06		46
=	07		47
sto	08		48
0	09		49
stop	10		50
×	11		51
rcl	12		52
0	13		53
=	14		54
goto	15		55
1	16		56
0	17		57
	18		58
	19		59
	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

DISCOUNT

KEY	#	KEY	#
HALT	00		40
+/−	01		41
÷	02		42
1	03		43
0	04		44
0	05		45
+	06		46
1	07		47
=	08		48
sto	09		49
0	10		50
stop	11		51
×	12		52
rcl	13		53
0	14		54
=	15		55
goto	16		56
1	17		57
1	18		58
	19		59
	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

Execution:

% discount/run/gross price/run/ discounted price/gross price/run/ discounted price

To enter new discount:

goto/0/1/new discount/run/

MARK-UP: GROSS PERCENTAGE GIVEN

Marks up by given % of marked up price (so, for instance, £90 + 10% = £100).

Execution:

%/run/old price/run/new price/
old price/run/new price/

To enter new %

goto/0/1/new %/run/

Example:

%	10/run
old price	90/run
new price	99.999
old price	45/run
new price	50

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
−	05		45
1	06		46
=	07		47
+/−	08		48
1/x	09		49
sto	10		50
0	11		51
stop	12		52
×	13		53
rcl	14		54
0	15		55
=	16		56
goto	17		57
1	18		58
2	19		59
	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

DISCOUNT OR TAX: PERCENTAGE OF NET SUM GIVEN

KEY	#	KEY	#
HALT	00		40
÷	01		41
(	02		42
+	03		43
1	04		44
0	05		45
0	06		46
)	07		47
=	08		48
sto	09		49
0	10		50
stop	11		51
—	12		52
(	13		53
×	14		54
rcl	15		55
0	16		56
=	17		57
stop	18		58
)	19		59
=	20		60
goto	21		61
1	22		62
1	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

Finds mark-up or tax and net price given the gross price.

Execution:

%/run/gross price/run/% discount/run/net price/gross price/run/% discount/run/net price/

To enter new %
goto/0/1/new %/run/

Example:

Percentage	8%/run
Gross price	£10/run
% discount	0.74/run
Net price	9.259

PERCENTAGE CHANGE ARISING FROM A CHANGE IN MARK-UP OR DISCOUNT.

Given a change of mark-up this program finds by what % of the present gross price the price of goods should be changed.

Execution:

Old mark-up/run/new mark-up/run/% change

Enter discounts as negative mark-ups.

Example:

old mark-up	25/run
new mark-up	12.5/run
percentage change	−10

KEY	#	KEY	#
HALT	00		40
sto	01		41
0	02		42
÷	03		43
1	04		44
0	05		45
0	06		46
+	07		47
1	08		48
=	09		49
1/x	10		50
×	11		51
(	12		52
stop	13		53
−	14		54
rcl	15		55
0	16		56
)	17		57
=	18		58
goto	19		59
0	20		60
0	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

VAT (1)

KEY	#	KEY	#
HALT	00		40
0	01		41
sto	02		42
1	03		43
sto	04		44
2	05		45
stop	06		46
÷	07		47
1	08		48
0	09		49
0	10		50
=	11		51
sto	12		52
0	13		53
stop	14		54
+	15		55
(	16		56
×	17		57
rcl	18		58
0	19		59
=	20		60
M+	21		61
1	22		62
)	23		63
=	24		64
M+	25		65
2	26		66
goto	27		67
1	28		68
4	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

Given several rates of vat and a collection of items priced vat exclusive, this program finds the vat inclusive price of each item, the total amount of vat and the grand total of the prices vat inclusive.

Pre-execution:

goto/0/1

Execution:

run/1st rate of vat/run/
price of item at 1st rate/run/
vat inclusive price/price of next item/
run/vat inclusive price/ . . . etc
goto/0/7/new rate of vat/run/price of item/run/vat inclusive price/
price of another item/run/vat inclusive price . . . etc.
To obtain grand total
rcl/1/total vat
rcl/2/grand total vat inclusive prices

Example:

I have three items priced £50, £30.50 and £42.24 vat exclusive with vat at 8% and two items priced £16.32 and £24.56 vat exclusive at 12½%:
run/8/run/50/run/54/30.50/
run/32.94/42.25/run/45.63/
goto/0/7/12.5/run/16.32/
run/18.36/24.56/run/27.63
rcl/1/14.93
rcl/2/178.56
So the vat inclusive prices were: £54, £32.94, £45.63, £16.32 and £27.63 respectively.
The total amount of vat was £14.93 and the grand total vat inclusive was £178.56.

VAT (2)

This program does the same as VAT (1) except that the rate of vat must be entered for each item – it is to be used when the items have not already been sorted into batches at each vat rate.

Pre-execution:

goto/0/1/run

Execution:

Price of item/run/vat rate/run/vat inclusive price/price of another item/run/vat rate/run/vat inclusive price/price of item etc.
rcl/1/total vat
rcl/2/grand total of prices and tax

KEY	#	KEY	#
HALT	00		40
0	01		41
sto	02		42
1	03		43
sto	04		44
2	05		45
stop	06		46
+	07		47
(	08		48
×	09		49
stop	10		50
÷	11		51
1	12		52
0	13		53
0	14		54
=	15		55
M+	16		56
1	17		57
)	18		58
=	19		59
M+	20		60
2	21		61
goto	22		62
0	23		63
6	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

VAT (3)

KEY	#	KEY	#
HALT	00		40
0	01		41
sto	02		42
1	03		43
sto	04		44
2	05		45
stop	06		46
÷	07		47
1	08		48
0	09		49
0	10		50
+	11		51
1	12		52
=	13		53
sto	14		54
0	15		55
stop	16		56
÷	17		57
(	18		58
×	19		59
(	20		60
1	21		61
—	22		62
rcl	23		63
0	24		64
1/x	25		65
)	26		66
=	27		67
M+	28		68
1	29		69
rcl	30		70
0	31		71
)	32		72
=	33		73
M+	34		74
2	35		75
goto	36		76
1	37		77
6	38		78
	39		79

Given several rates of vat and a collection of items priced vat inclusive, this program finds the vat exclusive price of each item, the total amount of vat and the grand total of the prices vat exclusive.

Pre-execution:

goto/0/1/run/

Execution:

rate of vat/run/price of first item/run/vat exclusive price/price of second item/run/vat exclusive price/ . . .

goto/0/7/2nd rate of vat/price of first item at second rate/vat exclusive price/ . . . etc.

To obtain grand total:
rcl/1/total vat
rcl/2/grand total vat exclusive prices

VAT (4)

This program does the same as VAT (3) except that the vat rate must be entered for each item. It is to be used when the items have not been sorted into batches according to vat rate.

Pre-execution:

goto/0/1/run

Execution:

vat inclusive price/run/vat rate/run/vat exclusive price/vat inclusive price/run/vat rate/run/vat exclusive price/ etc.

To obtain totals:
rcl/1/total vat
rcl/2/grand total vat exclusive

KEY	#	KEY	#
HALT	00		40
0	01		41
sto	02		42
1	03		43
sto	04		44
2	05		45
stop	06		46
−	07		47
(	08		48
÷	09		49
(	10		50
stop	11		51
÷	12		52
1	13		53
0	14		54
0	15		55
+	16		56
1	17		57
)	18		58
=	19		59
M+	20		60
2	21		61
sto	22		62
0	23		63
)	24		64
=	25		65
M+	26		66
1	27		67
rcl	28		68
0	29		69
goto	30		70
0	31		71
6	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

6. FINANCE: COMPOUND INTEREST

Single sum

$$A = P\left(1 + \frac{r}{100}\right)^n$$

A is the amount
P the principal
r the interest % per period
n the number of periods

Two sets of programs are given – one in which any period may be used and one in which the period is the year, but the interest is compounded six monthly at half the annual rate (as is the practice of some financial institutions).

Program 6.1 : finds A
6.3 : finds n
6.5 : finds r
6.7 : finds P
Program 6.2 : finds A with interest compounded six monthly
6.4 : finds n with interest compounded six monthly
6.6 : finds r with interest compounded six monthly
6.8 : finds P with interest compounded six monthly

SINGLE REPAYMENT LOAN (INTEREST COMPOUNDED EACH ACCOUNTING PERIOD)

Given:

Rate of interest per accounting period

Number of accounting periods

Initial sum

Find:

Final sum

Execution:

rate of interest/run/number of periods/run/amount of loan/run/amount of repayment

Example:

I borrow £100 at 10% per annum to be repaid at the end of three years – how much do I pay?

10/run/3/run/100/run/133.1

Answer £133.10

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
+	05		45
1	06		46
y^x	07		47
stop	08		48
×	09		49
stop	10		50
=	11		51
goto	12		52
0	13		53
0	14		54
	15		55
	16		56
	17		57
	18		58
	19		59
	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

SINGLE REPAYMENT LOAN

KEY	#	KEY	#
HALT	00		40
÷	01		41
2	02		42
0	03		43
0	04		44
+	05		45
1	06		46
=	07		47
log	08		48
×	09		49
stop	10		50
×	11		51
2	12		52
=	13		53
a log	14		54
×	15		55
stop	16		56
=	17		57
goto	18		58
0	19		59
0	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

Execution:

rate of interest/run/term in years/run/amount/run/amount of repayment

(interest compounded every six months)

Example:

Rate 8/run

Term 5/run

Amount (initial sum) 570/run

Answer £843.739

SINGLE REPAYMENT LOAN

Execution:

rate/run/initial sum/run/final sum/run/term

Interest compounded every accounting period

Example:

Rate	10.38/run
Initial sum	100/run
Final sum	150/run
Term	= 4.1056

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
+	05		45
1	06		46
=	07		47
ln	08		48
sto	09		49
0	10		50
stop	11		51
÷	12		52
stop	13		53
=	14		54
1/x	15		55
ln	16		56
÷	17		57
rcl	18		58
0	19		59
=	20		60
goto	21		61
0	22		62
0	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

SINGLE REPAYMENT LOAN

KEY	#	KEY	#
HALT	00		40
÷	01		41
2	02		42
0	03		43
0	04		44
+	05		45
1	06		46
=	07		47
log	08		48
sto	09		49
0	10		50
stop	11		51
÷	12		52
stop	13		53
=	14		54
1/x	15		55
log	16		56
÷	17		57
rcl	18		58
0	19		59
÷	20		60
2	21		61
=	22		62
goto	23		63
0	24		64
0	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

Execution:

rate/run/initial sum/run/final sum/run/term in years

Interest compounded every six months

Example:

Rate	10.3979/run
Initial sum	100/run
Final sum	150/run
Term	= 4

SINGLE REPAYMENT LOAN

Execution:

initial sum/run/final sum/run/
term/run/rate of interest

Example:

Initial sum	100/run/
Final sum	150/run/
Term	4/run/
Rate of interest	= 10.066819

KEY	#	KEY	#
HALT	00		40
÷	01		41
stop	02		42
1/x	03		43
=	04		44
log	05		45
÷	06		46
stop	07		47
=	08		48
a log	09		49
−	10		50
1	11		51
×	12		52
1	13		53
0	14		54
0	15		55
=	16		56
goto	17		57
0	18		58
0	19		59
	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

SINGLE REPAYMENT LOAN

KEY	#	KEY	#
HALT	00		40
÷	01		41
stop	02		42
=	03		43
1/x	04		44
ln	05		45
÷	06		46
(	07		47
stop	08		48
×	09		49
2	10		50
)	11		51
=	12		52
e^x	13		53
—	14		54
1	15		55
×	16		56
2	17		57
0	18		58
0	19		59
=	20		60
stop	21		61
goto	22		62
0	23		63
1	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

Interest compounded every six months

Execution:

Initial sum/run/final sum/run/term/run/rate of interest

Example:

Initial sum	100/run
Final sum	150/run
Term	4/run

Equivalent interest rate = 10.3979

PRESENT VALUE OF A SINGLE FUTURE PAYMENT

Execution:

rate/run/term/run/amount/run/
present value

Example:

I want to save some money now to grow into £1000 at 0.8% per month over 15 months: how much should I save?
0.8/run/15/run/1000/run/887.34
Answer = £887.34

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
+	05		45
1	06		46
y^x	07		47
stop	08		48
=	09		49
1/x	10		50
×	11		51
stop	12		52
=	13		53
goto	14		54
0	15		55
0	16		56
	17		57
	18		58
	19		59
	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

PRESENT VALUE OF A SINGLE FUTURE PAYMENT

Interest compounded every six months

Execution:

rate/run/term/run/amount/run/
present value

Example:

Rate	14/run
Term	4/run
Amount	5000/run
Present value = £2910	

KEY	#	KEY	#
HALT	00		40
÷	01		41
2	02		42
0	03		43
0	04		44
+	05		45
1	06		46
=	07		47
1/x	08		48
log	09		49
×	10		50
sto	11		51
0	12		52
stop	13		53
×	14		54
2	15		55
=	16		56
a log	17		57
×	18		58
stop	19		59
=	20		60
goto	21		61
0	22		62
0	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

7. FINANCE: COMPOUND INTEREST

Regular Payments

It is assumed the payment is made at the end of each period, and that interest is compounded each period.

x = regular payment
r = interest % per period
n = number of periods

Investment:

Balance of savings = B

$$B = \frac{100x}{r}\left(1+\frac{r}{100}\right)^n - \frac{100x}{r}$$

Program 7.1 finds B
7.2 finds x
7.3 finds n
7.4 finds r

Loans:

Amount, A, to be paid off completely

$$A = \frac{100x}{r}\left[1-\left(1+\frac{r}{100}\right)^{-n}\right]$$

Program 7.5 finds x
7.6 finds n
7.7 finds A
7.8 finds r

Balance of a loan:

$$B = A\left(1+\frac{r}{100}\right)^n - \frac{100x}{r}\left[\left(1+\frac{r}{100}\right)^n + 1\right]$$

Program 9 finds B given the number of payments already made

To find B given the number of payments still to be made use program 7.

"True Annual Interest Rate":

Program 10 finds the true annual interest rate using an approximate formula sometimes used in normal accounting practice.

To find the real true annual interest rate use program 8 and the "period rate to annual rate program" in Section 8.

Futures:

Program 11 finds the present value of a series of unequal future payments.

To find the present value of a series of equal future payments use program 7.

Program 12 finds the present value of a series of equal future payments followed by a single payment.

BALANCE OF SAVINGS

Execution:

Interest rate/run/number of periods/run/payment/run/balance

Example:

If I save £200 each year at 8% per annum how much will I have at the end of five years (not allowing for inflation).

8/run/5/run/200/run/1173.3202
Answer £1173.32

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
+	05		45
sto	06		46
0	07		47
1	08		48
y^x	09		49
stop	10		50
—	11		51
1	12		52
×	13		53
stop	14		54
÷	15		55
rcl	16		56
0	17		57
=	18		58
goto	19		59
0	20		60
0	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

INVESTMENT PLANNING

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
+	05		45
sto	06		46
0	07		47
1	08		48
y^x	09		49
stop	10		50
−	11		51
1	12		52
=	13		53
1/x	14		54
×	15		55
stop	16		56
×	17		57
rcl	18		58
0	19		59
=	20		60
goto	21		61
0	22		62
0	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

Execution:

Interest rate/run/number of periods/run/amount/run/payment

Example:

I wish to save £2000 over 18 months. Interest is 0.8 per cent per month – how much should I save each month?

0.8/run/18/run/2000/run/103.746

Answer: £103.75 per month

INVESTMENT PLANNING

Execution:

rate of interest/run/amount/run/payment/run/number of periods

Example:

I wish to save £2000 paying £50 per month at 0.8% per month interest. How long will it take?

0.8/run/2000/run/50/run/
34.842598

Answer: 35 months

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
×	05		45
sto	06		46
0	07		47
stop	08		48
÷	09		49
stop	10		50
+	11		51
1	12		52
=	13		53
log	14		54
÷	15		55
(	16		56
rcl	17		57
0	18		58
+	19		59
1	20		60
)	21		61
log	22		62
=	23		63
goto	24		64
0	25		65
0	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

INTEREST ON SAVINGS

KEY	#	KEY	#
HALT	00	×	40
sto	01	rcl	41
3	02	3	42
stop	03	−	43
sto	04	(	44
6	05	1	45
stop	06	+	46
sto	07	rcl	47
5	08	2	48
1	09	y^x	49
sto	10	rcl	50
2	11	5	51
1	12	−	52
./EE	13	1	53
./EE	14	×	54
5	15	rcl	55
+/−	16	6	56
sto	17	)	57
1	18	=	58
rcl	19	gin	59
2	20	1	60
sto	21	9	61
0	22	+/−	62
goto	23	gin	63
3	24	2	64
0	25	6	65
rcl	26	rcl	66
2	27	2	67
sto	28	×	68
1	29	1	69
rcl	30	0	70
0	31	0	71
+	32	=	72
rcl	33	goto	73
1	34	0	74
÷	35	0	75
2	36		76
=	37		77
sto	38		78
2	39		79

Execution:

Balance of investment/run/regular payment/run/number of periods/run/interest rate

Example:

I have saved £600 having paid £100 a year for five years. What rate of interest am I being paid?

600/run/100/run/5/run/9.1280625

Answer: 9.13% per annum

Warning:

This program may take several minutes to execute.

Remember these programs assume payments are made at the **end** of each period.

REGULAR REPAYMENT LOAN

Execution:

rate/run/term/run/amount/run/ regular repayment

Example:

Rate	1/run
Term	36/run
Amount	100/run
Regular repayment =	£3.32

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
+	05		45
sto	06		46
0	07		47
1	08		48
=	09		49
log	10		50
×	11		51
stop	12		52
=	13		53
a log	14		54
1/x	15		55
—	16		56
1	17		57
=	18		58
+/—	19		59
÷	20		60
rcl	21		61
0	22		62
÷	23		63
stop	24		64
=	25		65
1/x	26		66
goto	27		67
0	28		68
0	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

REGULAR REPAYMENT LOAN

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
×	05		45
sto	06		46
0	07		47
stop	08		48
÷	09		49
stop	10		50
−	11		51
1	12		52
=	13		53
+/−	14		54
1/x	15		55
ln	16		56
÷	17		57
(	18		58
rcl	19		59
0	20		60
+	21		61
1	22		62
=	23		63
ln	24		64
)	25		65
=	26		66
goto	27		67
0	28		68
0	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

Execution:

rate/run/amount of loan/run/repayment/run/number of repayments

Example:

Rate	10/run
Initial sum	1000/run
Annual repayment	250/run
Answer 5.3596 years	

PRESENT VALUE OF A SERIES OF EQUAL FUTURE PAYMENTS

Execution:

rate/run/number of payments/run/amount of each payment/run/present value

Example:

rate	13/run
No. of payments	19/run
Amount of each payment	10,000/run
Present Value	£69,379

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
+	05		45
sto	06		46
0	07		47
1	08		48
=	09		49
ln	10		50
×	11		51
stop	12		52
=	13		53
+/−	14		54
e^x	15		55
−	16		56
1	17		57
=	18		58
+/−	19		59
÷	20		60
rcl	21		61
0	22		62
×	23		63
stop	24		64
=	25		65
goto	26		66
0	27		67
0	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

INTEREST ON LOAN

KEY	#	KEY	#
HALT	00	=	40
×	01	x^2	41
1	02	—	42
0	03	1	43
0	04	./EE	44
÷	05	./EE	45
stop	06	8	46
=	07	+/—	47
sto	08	=	48
0	09	+/—	49
sto	10	gin	50
1	11	5	51
stop	12	8	52
+/—	13	rcl	53
sto	14	3	54
2	15	goto	55
rcl	16	0	56
1	17	0	57
÷	18	rcl	58
1	19	3	59
0	20	sto	60
0	21	1	61
+	22	goto	62
1	23	1	63
y^x	24	6	64
rcl	25		65
2	26		66
=	27		67
+/—	28		68
+	29		69
1	30		70
×	31		71
rcl	32		72
0	33		73
=	34		74
sto	35		75
3	36		76
—	37		77
rcl	38		78
1	39		79

Execution:

Regular payment/run/amount of loan/run/number of payments/run/interest rate per period

Example:

I borrow £150 and pay £16.45 for fifteen months: what is the interest?

16.45/run/150/run/15/run/6.9816616

Answer: An incredible 6.98% per month

Warning:

This program may take several minutes to run.

REGULAR REPAYMENT LOAN

Execution:

rate/run/number of repayments/ run/repayment/run/original amount/run/balance

Example:

Rate	9/run
No. of payments	5/run
payment	100/run
Original amount	500/run
Balance = £170.84	

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
+	05		45
sto	06		46
0	07		47
1	08		48
=	09		49
log	10		50
×	11		51
stop	12		52
=	13		53
a log	14		54
×	15		55
(	16		56
stop	17		57
÷	18		58
rcl	19		59
0	20		60
=	21		61
sto	22		62
0	23		63
+/−	24		64
+	25		65
stop	26		66
)	27		67
+	28		68
rcl	29		69
0	30		70
=	31		71
goto	32		72
0	33		73
0	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

APPROXIMATE TRUE ANNUAL INTEREST RATE

KEY	#	KEY	#
HALT	00	0	40
sto	01	0	41
0	02	=	42
×	03	goto	43
stop	04	0	44
÷	05	0	45
stop	06		46
—	07		47
1	08		48
=	09		49
1/x	10		50
×	11		51
(	12		52
rcl	13		53
0	14		54
+	15		55
1	16		56
)	17		57
×	18		58
3	19		59
+	20		60
rcl	21		61
0	22		62
—	23		63
(	24		64
stop	25		65
sto	26		66
0	27		67
×	28		68
3	29		69
)	30		70
+	31		71
2	32		72
=	33		73
1/x	34		74
×	35		75
rcl	36		76
0	37		77
×	38		78
6	39		79

Execution:

number of repayments/run/ amount of repayment/run/ amount of loan/run/number of payments per year/run/ interest rate per annum

Example:

I borrow £150 and pay back £16.45 per month for 15 months: what is the rate of interest?
15/run/16.45/run/150/run/12/ run/ 129.9202
Answer: An incredible 129.92% p.a.

Note: this program is not any use for long term loans.

PRESENT VALUE OF A SERIES OF POSSIBLY UNEQUAL FUTURE PAYMENTS

Execution:

Interest rate/run/p_n/run/p_{n-1}/ . . . . . ./p_1/run/present value

where the payments are p_1 at end of first year, p_2 at end of second year etc.

To start again:
goto/0/1

Example:

rate	14/run
1982	20000/run
1981	20000/run
1980	15000/run
1979	12000/run
1978	10000/run

Present value (1977): £50,359

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
+	05		45
1	06		46
=	07		47
1/x	08		48
sto	09		49
0	10		50
stop	11		51
×	12		52
rcl	13		53
0	14		54
+	15		55
goto	16		56
1	17		57
1	18		58
	19		59
	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

PRESENT VALUE OF A SERIES OF EQUAL PAYMENTS FOLLOWED BY A SINGLE PAYMENT

KEY	#	KEY	#
HALT	00	0	40
÷	01	0	41
1	02		42
0	03		43
0	04		44
+	05		45
sto	06		46
0	07		47
1	08		48
=	09		49
ln	10		50
×	11		51
stop	12		52
=	13		53
+/—	14		54
e^x	15		55
sto	16		56
1	17		57
rcl	18		58
0	19		59
1/x	20		60
×	21		61
stop	22		62
=	23		63
sto	24		64
0	25		65
rcl	26		66
1	27		67
×	28		68
(	29		69
stop	30		70
—	31		71
rcl	32		72
0	33		73
)	34		74
+	35		75
rcl	36		76
0	37		77
=	38		78
goto	39		79

Execution:

Interest rate/run/number of periods/run/regular payment/run/final payment/run/present value

Example:

Rate	6.5/run
No. of payments	19/run
regular payment	35/run
final payment	1000/run
present value £677.96	

8. FINANCE: INTEREST RATE CONVERSIONS

8.1 Annual rate to period rate
8.2 Period rate to annual rate
8.3 Period rate to period rate

ANNUAL RATE TO PERIOD RATE

Given the annual rate of interest and the number of periods per year, this program finds the true rate of interest per period.

Execution:

Annual rate/run/number of periods/run/period rate

Example:

A savings account pays interest monthly at 10% per annum — what is the rate of interest per month:

10/run/12/run/7.9741–01

Answer 0.797%

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
+	05		45
1	06		46
y^x	07		47
stop	08		48
1/x	09		49
—	10		50
1	11		51
×	12		52
1	13		53
0	14		54
0	15		55
=	16		56
goto	17		57
0	18		58
0	19		59
	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

PERIOD RATE TO ANNUAL RATE

KEY	#	KEY	#
HALT	00		40
×	01		41
(	02		42
stop	03		43
÷	04		44
1	05		45
0	06		46
0	07		47
+	08		48
1	09		49
=	10		50
log	11		51
)	12		52
=	13		53
a log	14		54
−	15		55
1	16		56
×	17		57
1	18		58
0	19		59
0	20		60
=	21		61
goto	22		62
0	23		63
0	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

Execution:

number of periods per year/run/
period rate/run/annual rate

Example:

Interest is 1.9% per month – what is it per year?

12/run
1.9/run
Annual rate = 25.34%

PERIOD RATE TO PERIOD RATE

Given the period rate of interest and the number of periods per year, this program finds rate of interest per a different period.

Execution:

period rate/run/no. of periods per year/run/no. of new periods per year/new period rate

Example:

A credit card company charges 1.75% interest per month but adds interest daily – what is the daily interest rate?

There are twelve months to the year and 365 days so:
1.75/run/12/run/365/run/
5.7052–02
Answer: 0.057052%

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
+	05		45
1	06		46
y^x	07		47
(	08		48
stop	09		49
÷	10		50
stop	11		51
)	12		52
−	13		53
1	14		54
×	15		55
1	16		56
0	17		57
0	18		58
=	19		59
goto	20		60
0	21		61
0	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

9. MORTGAGES

Program 9.1 finds the amount of repayment

9.2 the term

9.3 the balance, given the amount paid so far

9.4 the balance, given the number of years left to pay

9.5 the tax relief

To find the interest rate on a mortgage multiply the monthly repayment by 12 to find the annual repayment, then use program 8 of section 7, where the period is one year.

MORTGAGE REPAYMENTS

Execution:

Interest rate/run/term/run/ amount/run/monthly repayment

Example:

The balance to pay on my mortgage is £4270 and the present rate of interest is 11%, it has twelve years to run. What is the monthly repayment?

11/run/12/run/4270/run/54.81

Answer: £54.81 per month

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
+	05		45
sto	06		46
0	07		47
1	08		48
y^x	09		49
stop	10		50
=	11		51
1/x	12		52
−	13		53
1	14		54
÷	15		55
rcl	16		56
0	17		57
+/−	18		58
÷	19		59
stop	20		60
=	21		61
1/x	22		62
÷	23		63
1	24		64
2	25		65
=	26		66
goto	27		67
0	28		68
0	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

MORTGAGE TERM

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
×	05		45
sto	06		46
0	07		47
stop	08		48
÷	09		49
stop	10		50
÷	11		51
1	12		52
2	13		53
—	14		54
1	15		55
=	16		56
+/—	17		57
1/x	18		58
log	19		59
÷	20		60
(	21		61
rcl	22		62
0	23		63
+	24		64
1	25		65
=	26		66
log	27		67
)	28		68
=	29		69
goto	30		70
0	31		71
0	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

Execution:

rate/run/amount/run/monthly payment/run/mortgage term

Example:

rate	11/run
amount of mortgage	7000/run
repayment	80/run

Result is 15.5223

BALANCE OF A MORTGAGE

Execution:

interest rate/run/number of years since mortgage taken out/run/monthly repayment/run/original amount/run/balance outstanding

Example:

I bought a house seven years ago and took out a mortgage for £5500 at 11$\frac{1}{2}$%. My monthly repayment has been £70 – how much do I still owe?

11.5/run/7/run/70/run/5500/run/
3438

Answer: £3438

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
+	05		45
sto	06		46
0	07		47
1	08		48
y^x	09		49
stop	10		50
×	11		51
(	12		52
stop	13		53
×	14		54
1	15		55
2	16		56
÷	17		57
rcl	18		58
0	19		59
=	20		60
sto	21		61
0	22		62
+/−	23		63
+	24		64
stop	25		65
)	26		66
+	27		67
rcl	28		68
0	29		69
=	30		70
goto	31		71
0	32		72
0	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

BALANCE OF A MORTGAGE

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
+	05		45
sto	06		46
0	07		47
1	08		48
y^x	09		49
stop	10		50
=	11		51
1/x	12		52
+/−	13		53
+	14		54
1	15		55
×	16		56
stop	17		57
×	18		58
1	19		59
2	20		60
÷	21		61
rcl	22		62
0	23		63
=	24		64
goto	25		65
0	26		66
0	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

Execution:

Interest rate/run/number of years to run/run/monthly payment/run/balance

Example:

My mortgage has twelve years to run, my present monthly payment is £50 and the interest rate is $10\frac{1}{2}$%. What is the balance outstanding?

10.5/run/12/run/50/run/3989.97
Answer: £3990

TAX RELIEF ON MORTGAGE

Execution:

balance/run/interest rate/run/ standard rate of tax/run/tax relief

Example:

Balance	6000/run
rate	10.75/run
standard rate of tax	35/run

Tax relief = £225.75

KEY	#	KEY	#
HALT	00		40
×	01		41
stop	02		42
×	03		43
(	04		44
stop	05		45
÷	06		46
1	07		47
0	08		48
0	09		49
0	10		50
0	11		51
)	12		52
=	13		53
goto	14		54
0	15		55
0	16		56
	17		57
	18		58
	19		59
	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

10. BUSINESS METHODS

Program 11 of section 7 can be used for discounted cash flow, but a program more specifically designed for this application is given here.

For sinking funds, programs 1-4 of section 7 can be used.

10.1 Discounted cash flow
10.2 Learning curve-finds prices
10.3 Learning curve-finds learning

DISCOUNTED CASH FLOW

Execution:

interest rate/run/cumulative DCF up to year 0 (this will usually be zero)/run/cash flow for year 0/run/cumulative DCF up to and including year zero/cash flow for year 1/run/cumulative DCF up to and including year 1/cash flow for year 2/run/cumulative DCF up to and including year 2/cash flow for year 3/run/cumulative . . . etc.

To start again:
goto/0/1

KEY	#	KEY	#
HALT	00		40
÷	01		41
1	02		42
0	03		43
0	04		44
+	05		45
1	06		46
=	07		47
sto	08		48
0	09		49
stop	10		50
sto	11		51
1	12		52
1	13		53
sto	14		54
2	15		55
rcl	16		56
1	17		57
+	18		58
(	19		59
stop	20		60
×	21		61
rcl	22		62
2	23		63
)	24		64
=	25		65
sto	26		66
1	27		67
rcl	28		68
2	29		69
÷	30		70
rcl	31		71
0	32		72
=	33		73
sto	34		74
2	35		75
goto	36		76
1	37		77
6	38		78
	39		79

LEARNING CURVE (FIND PRICES)

KEY	#	KEY	#
HALT	00		40
y^x	01		41
(	02		42
stop	03		43
log	04		44
÷	05		45
2	06		46
log	07		47
=	08		48
sto	09		49
0	10		50
)	11		51
×	12		52
stop	13		53
=	14		54
stop	15		55
×	16		56
(	17		57
1	18		58
+	19		59
rcl	20		60
0	21		61
)	22		62
=	23		63
goto	24		64
0	25		65
0	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

P_1 = price of first item

P_n = (marginal) price of nth item

$\bar{P}_n$ = Cumulative average price of first n items

L = Learning

$P_n = P_1 n^{\log_2 L}(1 + \log_2 L)$

$\bar{P}_n = p_1 n^{\log_2 L}$

Execution:

n/run/L/run/P_1/run/$\bar{P}_n$/run/P_n

LEARNING CURVE (FIND LEARNING)

Execution:

(a) P_n/run/$\bar{P}_n$/run/L

(b) goto/1/5/
$\bar{P}_n$/run/P_1/run/n/run/L

KEY	#	KEY	#
HALT	00		40
÷	01		41
stop	02		42
−	03		43
1	04		44
=	05		45
y^x	06		46
2	07		47
x ◂▸ y	08		48
=	09		49
goto	10		50
0	11		51
0	12		52
	13		53
	14		54
÷	15		55
stop	16		56
=	17		57
log	18		58
÷	19		59
stop	20		60
log	21		61
=	22		62
y^x	23		63
2	24		64
x ◂▸ y	25		65
=	26		66
goto	27		67
0	28		68
0	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

11. STATISTICAL SAMPLING

11.1 Mean variance and standard deviation

11.2 Sample mean and standard deviation

11.3 Normalisation of sample

11.4 Correlation coefficient

11.5 Linear regression

MEAN, VARIANCE AND STANDARD DEVIATION

Mean, $\mu = \frac{1}{n} \sum_{i=1}^{n} x_i$

Variance, $\sigma^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i - \mu)^2$

Standard deviation, $\sigma = \sqrt{\text{variance}}$

Execution:

goto/0/1/run/x_1/
run/x_2/run/x_3/
run/ . . . x_n/run/
goto/2/0/run/μ
/run/σ^2/run/σ/
x_{n+1}/run/x_{n+2} . . .
etc.

Program gives mean, variance and standard deviation to date. To start again with another sample:

goto/0/1/
run/x_1/run/x_2 . . . etc.

To recall number of x's entered in sample:
/rcl/2/.

KEY	#	KEY	#
HALT	00	goto	40
0	01	0	41
sto	02	8	42
0	03		43
sto	04		44
1	05		45
sto	06		46
2	07		47
stop	08		48
M+	09		49
0	10		50
x^2	11		51
M+	12		52
1	13		53
1	14		54
M+	15		55
2	16		56
goto	17		57
0	18		58
8	19		59
rcl	20		60
0	21		61
÷	22		62
rcl	23		63
2	24		64
=	25		65
stop	26		66
x^2	27		67
+/−	28		68
+	29		69
(	30		70
rcl	31		71
1	32		72
÷	33		73
rcl	34		74
2	35		75
)	36		76
=	37		77
stop	38		78
$\sqrt{x}$	39		79

SAMPLE MEAN AND STANDARD DEVIATION (ESTIMATE OF STANDARD DEVIATION)

KEY	#	KEY	#
HALT	00	2	40
0	01	—	41
sto	02	1	42
0	03	)	43
sto	04	=	44
1	05	$\sqrt{x}$	45
sto	06	goto	46
2	07	0	47
stop	08	8	48
M+	09		49
0	10		50
x^2	11		51
M+	12		52
1	13		53
1	14		54
M+	15		55
2	16		56
rcl	17		57
2	18		58
goto	19		59
0	20		60
8	21		61
rcl	22		62
0	23		63
÷	24		64
rcl	25		65
2	26		66
=	27		67
stop	28		68
x^2	29		69
×	30		70
rcl	31		71
2	32		72
+/−	33		73
+	34		74
rcl	35		75
1	36		76
÷	37		77
(	38		78
rcl	39		79

Mean, $m = \frac{1}{n}\sum_{i=1}^{n} x_i$

Standard deviation

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - m)^2}$$

Execution:

goto/0/1/run/x_1/run/x_2/run/x_3/run/ . . . x_n/run/goto/2/2/run/m/run/s/x_{n+1}/run/x_{n+2} . . . etc.

Program gives mean and standard deviation to date. To start again with a new sample:

goto/0/1/run/x_1/run etc.

The display shows number of x's entered so far after each x is entered

NORMALISATION

Convert to mean 0, s.d.1

$$X_i = \frac{x_i - \bar{x}}{\sigma_x}$$

x_1/run/x_2/run/. . .
. . ./x_n/run/goto/
2/1/run/x_j/run/
X_j/new x_j/run/
new X_j/. . . etc.

to recover $\bar{x}$: rcl 1
to recover σ_x : rcl 2

KEY	#	KEY	#
HALT	00	$\sqrt{x}$	40
sto	01	sto	41
1	02	2	42
x^2	03	stop	43
sto	04	−	44
2	05	rcl	45
1	06	1	46
sto	07	−	47
0	08	rcl	48
stop	09	2	49
M+	10	=	50
1	11	goto	51
x^2	12	4	52
M+	13	3	53
2	14		54
1	15		55
M+	16		56
0	17		57
goto	18		58
0	19		59
9	20		60
rcl	21		61
1	22		62
÷	23		63
rcl	24		64
0	25		65
=	26		66
sto	27		67
1	28		68
x^2	29		69
+/−	30		70
+	31		71
(	32		72
rcl	33		73
2	34		74
÷	35		75
rcl	36		76
0	37		77
)	38		78
=	39		79

CORRELATION COEFFICIENT

KEY	#	KEY	#
HALT	00	rcl	40
sto	01	4	41
0	02	÷	42
sto	03	rcl	43
1	04	0	44
sto	05	÷	45
2	06	rcl	46
sto	07	2	47
3	08	—	48
sto	09	rcl	49
4	10	5	50
sto	11	1/x	51
5	12	÷	52
stop	13	(	53
M+	14	rcl	54
0	15	1	55
sto	16	÷	56
6	17	rcl	57
×	18	0	58
stop	19	x^2	59
=	20	—	60
M+	21	rcl	61
4	22	5	62
x◂▸y	23	1/x	63
M+	24	×	64
2	25	(	65
x^2	26	rcl	66
M+	27	3	67
3	28	÷	68
rcl	29	rcl	69
6	30	2	70
x^2	31	x^2	71
M+	32	—	72
2	33	rcl	73
1	34	5	74
M+	35	1/x	75
5	36	)	76
goto	37	)	77
1	38	$\sqrt{x}$	78
3	39	=	79

$$r = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\{(\sum_{i=1}^{n} (x_i - \bar{x})^2)(\sum_{i=1}^{n} (y_i - \bar{y})^2)\}}}$$

Where $\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i \quad \bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$

Execution:

0/run/x_1/run/y_1/run/
x_2/run/y_2/run/. . .
. . . /x_n/run/y_n/run/
goto/4/0/run/r

Test

0/run/1/run/2/run/3/
run/4/run/5/run/6/
run/7/run/8/run/goto/
4/0/run/1

LINEAR REGRESSION

Regression line
$y = mx + c$
(y on x)
slope,

$$m = \frac{\sum_{i=1}^{n} (x_i - \overline{x})(y_i - \overline{y})}{\sum_{i=1}^{n} (x_i - \overline{x})^2}$$

intercept,

$$c = \overline{y} - m\overline{x}$$

where $\overline{x} = \frac{1}{n} \sum_{i=1}^{n} x_i \quad \overline{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$

Execution:

0/run/x_1/run/y_1/run/
. . . x_n/run/y_n/run/
goto/3/4/run/m/run/c

Test:

0/run/1/run/2
run/3/run . . . . /7/run/
8/run/goto/3/4/run/
/run/1

KEY	#	KEY	#
HALT	00	(	40
sto	01	rcl	41
0	02	0	42
sto	03	×	43
1	04	rcl	44
sto	05	2	45
2	06	)	46
sto	07	÷	47
3	08	(	48
sto	09	rcl	49
4	10	1	50
stop	11	×	51
M+	12	rcl	52
0	13	4	53
sto	14	−	54
5	15	rcl	55
×	16	0	56
stop	17	x^2	57
M+	18	)	58
2	19	=	59
=	20	stop	60
M+	21	×	61
3	22	rcl	62
rcl	23	0	63
5	24	+/−	64
x^2	25	+	65
M+	26	rcl	66
1	27	2	67
1	28	÷	68
M+	29	rcl	69
4	30	4	70
goto	31	=	71
1	32	goto	72
1	33	0	73
rcl	34	0	74
3	35		75
×	36		76
rcl	37		77
4	38		78
−	39		79

12. SIGNIFICANCE TESTS

12.1 Hypothesis of zero correlation
12.2 Regression lines slope
12.3 Student's t-Test
12.4 χ^2 test
12.5 χ^2 test with Yates' correction
12.6 Two sample χ^2 test
12.7 Two sample χ^2 test with Yates' correction
12.8 z-statistic
12.9 Spearman's rank correlation coefficient

TESTING HYPOTHESIS OF ZERO CORRELATION

Execution:

If r is the sample correlation coefficient and N the sample size, large values of t suggest that the true correlation coefficient, ρ is non-zero.

r/run/N/run/t

$$t = r\frac{\sqrt{N-2}}{\sqrt{1-r^2}}$$

Test:

0.6/run/6/run/1.5

KEY	#	KEY	#
HALT	00		40
sto	01		41
0	02		42
stop	03		43
—	04		44
2	05		45
=	06		46
$\sqrt{x}$	07		47
×	08		48
rcl	09		49
0	10		50
÷	11		51
(	12		52
1	13		53
—	14		54
rcl	15		55
0	16		56
x^2	17		57
)	18		58
$\sqrt{x}$	19		59
=	20		60
goto	21		61
0	22		62
0	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

REGRESSION LINE SLOPE

Test whether the regression line slope, b is significantly different from b_0.

r = correlation coefficient,
N = sample size.

$$t = \frac{(b - b_0)\sqrt{N-2}}{\sqrt{1-r^2}}$$

Execution:

b_0/run/b/run/r/run/N/run/t

Test:

1.4/run/2/run/0.6/run/6/run/1.5

KEY	#	KEY	#
HALT	00		40
+/−	01		41
+	02		42
stop	03		43
×	04		44
(	05		45
stop	06		46
sto	07		47
0	08		48
stop	09		49
−	10		50
2	11		51
)	12		52
$\sqrt{x}$	13		53
÷	14		54
(	15		55
1	16		56
−	17		57
rcl	18		58
0	19		59
x^2	20		60
)	21		61
$\sqrt{x}$	22		62
=	23		63
goto	24		64
0	25		65
0	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

SAMPLE MEAN SAMPLE VARIANCE

STUDENT'S t TEST FOR MEAN m.

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

$$s^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2$$

$$t = \frac{(\bar{x} - m)\sqrt{n}}{s}$$

Execution:

x_1/run/x_2/run/ . . .
x_n/run/goto/2/1/
run/$\bar{x}$/run/s^2/run/m/run/t/

KEY	#	KEY	#
HALT	00	rcl	40
sto	01	2	41
0	02	−	42
x^2	03	1	43
sto	04	)	44
1	05	=	45
1	06	stop	46
sto	07	÷	47
2	08	rcl	48
stop	09	2	49
M+	10	=	50
0	11	$\sqrt{x}$	51
x^2	12	1/x	52
M+	13	×	53
1	14	(	54
1	15	rcl	55
M+	16	0	56
2	17	−	57
goto	18	stop	58
0	19	)	59
9	20	=	60
rcl	21	goto	61
0	22	0	62
÷	23	0	63
rcl	24		64
2	25		65
=	26		66
stop	27		67
sto	28		68
0	29		69
x^2	30		70
×	31		71
rcl	32		72
2	33		73
+/−	34		74
+	35		75
rcl	36		76
1	37		77
÷	38		78
(	39		79

χ^2 - STATISTIC

KEY	#	KEY	#
HALT	00		40
—	01		41
stop	02		42
sto	03		43
0	04		44
=	05		45
x^2	06		46
÷	07		47
rcl	08		48
0	09		49
)	10		50
+	11		51
(	12		52
goto	13		53
0	14		54
0	15		55
0	16		56
+	17		57
(	18		58
goto	19		59
0	20		60
0	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

Observed Values:

$x_1, x_2, \ldots$

Expected Values:

$e_1, e_2, \ldots$

$$\chi^2 = \sum_{i=1}^{n} \frac{(x_i - e_i)^2}{e_i}$$

Execution:

goto/1/6/run/x_1/run/e_1/run/x_2/run/e_2/run/. . ./x_n/run/e_n/run/ χ^2

After each e_i display shows χ^2 so far

Test:

goto/1/6/run/1/run/2/run/3/run . . ./7/run/8/run/ 1.0416667

x^2 WITH YATES' CORRECTION

Execution:

goto/2/2/run/
x_1/run/e_1/run . . .
. . . x_n/run/e_n/run/
x^2

Test:

goto/2/2/run/1/run/
2/run/3/run/ . . . /8/run/
2.6041–01

KEY	#	KEY	#
HALT	00		40
−	01		41
stop	02		42
sto	03		43
0	04		44
=	05		45
x^2	06		46
$\sqrt{x}$	07		47
−	08		48
./EE	09		49
5	10		50
=	11		51
x^2	12		52
÷	13		53
rcl	14		54
0	15		55
)	16		56
+	17		57
(	18		58
goto	19		59
0	20		60
0	21		61
0	22		62
+	23		63
(	24		64
goto	25		65
0	26		66
0	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

TWO SAMPLE χ^2

KEY	#	KEY	#
HALT	00		40
sto	01		41
0	02		42
−	03		43
stop	04		44
sto	05		45
1	06		46
=	07		47
x^2	08		48
÷	09		49
(	10		50
rcl	11		51
0	12		52
+	13		53
rcl	14		54
1	15		55
)	16		56
)	17		57
+	18		58
(	19		59
goto	20		60
0	21		61
0	22		62
0	23		63
+	24		64
(	25		65
goto	26		66
0	27		67
0	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

$$\chi^2 = \sum_{i=1}^{n} \frac{(x_i - y_i)^2}{x_i + y_i}$$

Execution:

goto/2/3/run/x_1/run/y_1/run/ . . . x_n/run/y_n run/χ^2

TWO SAMPLE CHI SQUARE TEST WITH YATES´ CORRECTION

Execution:

goto/2/8/run/x_1/run/
y_1/run/x_2/run/ . . .
. . . /x_n/run/y_n/
run/χ^2

KEY	#	KEY	#
HALT	00		40
sto	01		41
0	02		42
−	03		43
stop	04		44
sto	05		45
1	06		46
=	07		47
x^2	08		48
$\sqrt{x}$	09		49
−	10		50
1	11		51
=	12		52
x^2	13		53
÷	14		54
(	15		55
rcl	16		56
0	17		57
+	18		58
rcl	19		59
1	20		60
)	21		61
)	22		62
+	23		63
(	24		64
goto	25		65
0	26		66
0	27		67
0	28		68
+	29		69
(	30		70
goto	31		71
0	32		72
0	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

z-STATISTIC

KEY	#	KEY	#
HALT	00		40
sto	01		41
0	02		42
+/−	03		43
+	04		44
(	05		45
stop	06		46
÷	07		47
stop	08		48
sto	09		49
1	10		50
)	11		51
÷	12		52
(	13		53
rcl	14		54
0	15		55
−	16		56
rcl	17		57
0	18		58
x^2	19		59
÷	20		60
rcl	21		61
1	22		62
)	23		63
$\sqrt{x}$	24		64
=	25		65
goto	26		66
0	27		67
0	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

$$z = \frac{\frac{x}{n} - \theta}{\sqrt{\frac{\theta(1-\theta)}{n}}}$$

Execution:

θ/run/x/run/n/run/z

SPEARMAN´S RANK CORRELATION COEFFICIENT

$$\rho = 1 - 6 \frac{\sum_{i=1}^{n} (r_i - r_i')^2}{n^3 - n}$$

for pairs of ranks (r_i, r_i')

Execution:

r_1/run/r_1'/run/r_2/run/r_2'/run/ . . . . . . /r_n/run/r_n'/run/goto/2/3/run/ρ

KEY	#	KEY	#
HALT	00	1	40
−	01	=	41
stop	02	goto	42
=	03	0	43
x^2	04	0	44
sto	05		45
0	06		46
1	07		47
sto	08		48
1	09		49
stop	10		50
−	11		51
stop	12		52
=	13		53
x^2	14		54
M+	15		55
0	16		56
1	17		57
M+	18		58
1	19		59
goto	20		60
1	21		61
0	22		62
rcl	23		63
0	24		64
×	25		65
6	26		66
+/−	27		67
÷	28		68
rcl	29		69
1	30		70
÷	31		71
(	32		72
rcl	33		73
1	34		74
x^2	35		75
−	36		76
1	37		77
)	38		78
+	39		79

13. QUALITY CONTROL

13.1 Action and warning limits
13.2 Reliability of a parallel system

13. QUALITY CONTROL

13.1 Action and warning limits
13.2 Reliability of a parallel system

QUALITY CONTROL

Action and warning limits for proportion of batch having given attribute.

Typical values of α are
3.12 for action,
1.96 for warning.

$$a\pm \; = \; p \pm \alpha \; \sqrt{\frac{p(1-p)}{n}}$$

Execution:

p/run/n/run/α/run/
a−/run/a+

KEY	#	KEY	#
HALT	00		40
sto	01		41
0	02		42
×	03		43
(	04		44
+/−	05		45
+	06		46
1	07		47
)	08		48
÷	09		49
stop	10		50
=	11		51
$\sqrt{x}$	12		52
×	13		53
stop	14		54
=	15		55
sto	16		56
1	17		57
+/−	18		58
+	19		59
rcl	20		60
0	21		61
=	22		62
stop	23		63
rcl	24		64
1	25		65
+	26		66
rcl	27		67
0	28		68
=	29		69
stop	30		70
goto	31		71
0	32		72
1	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

RELIABILITY OF A PARALLEL SYSTEM

KEY	#	KEY	#
HALT	00		40
1	01		41
goto	02		42
1	03		43
1	04		44
+/−	05		45
+	06		46
1	07		47
×	08		48
rcl	09		49
0	10		50
=	11		51
sto	12		52
0	13		53
+/−	14		54
+	15		55
1	16		56
=	17		57
stop	18		58
goto	19		59
0	20		60
5	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

$$R = 1 - \prod_{i=1}^{n} (1 - R_i)$$

Execution:

run/R_1/run/R_2/
run/ . . . /R_n/run/R

Display shows reliability to date after each R_n is entered.

To use the program again:
goto/0/1

14. DISTRIBUTIONS

14.1 Normal distribution
14.2 Normal density function
14.3 % points of N (0, 1)
14.4 Poisson distribution

NORMAL DISTRIBUTION FUNCTION

$$\Phi(X) = \frac{1}{\sqrt{2\pi}} \int_{x}^{\infty} e^{-x^2/2}\, dx$$

To 4 or 5 sig. fig.

Execution:

X/run/Φ(X)

KEY	#	KEY	#
HALT	00	0	40
sto	01	7	41
1	02	0	42
×	03	8	43
./EE	04	×	44
2	05	rcl	45
3	06	0	46
1	07	−	47
6	08	./EE	48
4	09	1	49
2	10	4	50
+	11	2	51
1	12	2	52
=	13	4	53
1/x	14	8	54
sto	15	×	55
0	16	rcl	56
×	17	0	57
./EE	18	+	58
5	19	./EE	59
3	20	1	60
0	21	2	61
7	22	7	62
0	23	4	63
1	24	2	64
−	25	×	65
./EE	26	rcl	66
7	27	0	67
2	28	×	68
6	29	(	69
5	30	rcl	70
7	31	1	71
8	32	x^2	72
×	33	+/−	73
rcl	34	÷	74
0	35	2	75
+	36	=	76
./EE	37	e^x	77
7	38	)	78
1	39	=	79

NORMAL DENSITY FUNCTION

KEY	#	KEY	#
HALT	00		40
—	01		41
stop	02		42
÷	03		43
stop	04		44
sto	05		45
0	06		46
=	07		47
x^2	08		48
÷	09		49
2	10		50
+/—	11		51
=	12		52
e^x	13		53
÷	14		54
rcl	15		55
0	16		56
÷	17		57
(	18		58
π	19		59
×	20		60
2	21		61
=	22		62
$\sqrt{x}$	23		63
)	24		64
=	25		65
goto	26		66
0	27		67
0	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

$$\varphi = \frac{1}{\sigma\sqrt{2\pi}} \exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}$$

Execution:

x/run/μ/run/σ/run/φ

% POINTS OF N(O,1)

Given α, $0 < \alpha < 0.5$, finds x such that prob $(X > x) = \alpha$, where X is N(0, 1).

Execution:

α/run/X

KEY	#	KEY	#
HALT	00	1	40
×	01	./EE	41
1	02	0	42
./EE	03	0	43
0	04	6	44
0	05	=	45
0	06	goto	46
7	07	0	47
=	08	0	48
x^2	09		49
1/x	10		50
ln	11		51
$\sqrt{x}$	12		52
sto	13		53
0	14		54
×	15		55
4	16		56
+	17		57
1	18		58
2	19		59
./EE	20		60
5	21		61
÷	22		62
(	23		63
rcl	24		64
0	25		65
+	26		66
7	27		67
×	28		68
rcl	29		69
0	30		70
+	31		71
5	32		72
)	33		73
=	34		74
+/−	35		75
+	36		76
rcl	37		77
0	38		78
÷	39		79

POISSON DISTRIBUTION

KEY	#	KEY	#
HALT	00	goto	40
sto	01	1	41
0	02	5	42
stop	03	rcl	43
—	04	0	44
1	05	+/—	45
=	06	e^x	46
gin	07	×	47
5	08	rcl	48
5	09	2	49
sto	10	=	50
1	11	stop	51
1	12	goto	52
sto	13	0	53
2	14	1	54
rcl	15	rcl	55
2	16	0	56
×	17	+/—	57
rcl	18	e^x	58
0	19	goto	59
÷	20	0	60
(	21	0	61
rcl	22		62
1	23		63
+	24		64
1	25		65
)	26		66
=	27		67
sto	28		68
2	29		69
rcl	30		70
1	31		71
—	32		72
1	33		73
=	34		74
gin	35		75
4	36		76
3	37		77
sto	38		78
1	39		79

$$P(j) = \frac{e^{-\lambda}\lambda^j}{j!}$$

Execution:

λ/run/j/run/P(j)

15. TRANSFORMATIONS

15.1 Fisher's Z-transformation

15.2 Transforming χ^2 to normal

15.3 Transforming binomial to normal

FISHER´S Z TRANSFORMATION FOR CORRELATION COEFFICIENTS

$$Z = \tfrac{1}{2}\ln\left(\frac{1+\rho}{1-\rho}\right)$$

$$\sigma = \frac{1}{\sqrt{n-3}}$$

Z has approximately normal distribution:
n is the sample size and σ the standard deviation of Z

Execution:

ρ/run/Z/n/run/σ

KEY	#	KEY	#
HALT	00		40
+	01		41
1	02		42
÷	03		43
(	04		44
+/−	05		45
+	06		46
2	07		47
)	08		48
=	09		49
ln	10		50
÷	11		51
2	12		52
=	13		53
stop	14		54
−	15		55
3	16		56
=	17		57
$\sqrt{x}$	18		58
1/x	19		59
goto	20		60
0	21		61
0	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

TRANSFORMING χ^2 TO NORMAL

$$y = \sqrt{2x^2} - \sqrt{2n-1}$$

y has approximately normal distribution (N (0, 1)) when x has χ^2 distribution and a large number, n, degrees of freedom.

Execution:

Store n in memory 0

x/run/y

KEY	#	KEY	#
HALT	00		40
X^2	01		41
×	02		42
2	03		43
=	04		44
$\sqrt{x}$	05		45
—	06		46
(	07		47
2	08		48
×	09		49
rcl	10		50
0	11		51
—	12		52
1	13		53
)	14		54
$\sqrt{x}$	15		55
=	16		56
goto	17		57
0	18		58
0	19		59
	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

TRANSFORMING BINOMIAL TO NORMAL

$$Z = \frac{\frac{X}{n} - p}{\sqrt{\frac{p(1-p)}{n}}}$$

X Binomial, with parameters n, p. np, n(1 − p) both greater than 5. Then Z is approximately N(0, 1).

Execution:

p/run/n/run/
X/run/Z/new X/
run/new Z/ . . . etc.

To change p and n
goto/0/1/p/run/n/run

KEY	#	KEY	#
HALT	00		40
sto	01		41
0	02		42
×	03		43
stop	04		44
=	05		45
sto	06		46
1	07		47
×	08		48
(	09		49
1	10		50
−	11		51
rcl	12		52
0	13		53
)	14		54
=	15		55
1/x	16		56
$\sqrt{x}$	17		57
sto	18		58
0	19		59
×	20		60
rcl	21		61
1	22		62
=	23		63
sto	24		64
1	25		65
stop	26		66
×	27		67
rcl	28		68
0	29		69
−	30		70
rcl	31		71
1	32		72
=	33		73
goto	34		74
2	35		75
6	36		76
	37		77
	38		78
	39		79

16. RANDOM NUMBERS

16.1 Uniform on (0, 1)
16.2 From N (0, 1)
16.3 Uniform on (0, 1), alternative program

UNIFORMLY DISTRIBUTED RANDOM NUMBERS ON INTERVAL (0,1)

Execution:

Any number between 0 and 1/run/random No./run/new random No./run/. . .

KEY	#	KEY	#
HALT	00	goto	40
sto	01	0	41
0	02	3	42
rcl	03		43
0	04		44
×	05		45
3	06		46
3	07		47
3	08		48
4	09		49
7	10		50
—	11		51
(	12		52
+	13		53
1	14		54
./EE	15		55
./EE	16		56
9	17		57
—	18		58
1	19		59
./EE	20		60
./EE	21		61
9	22		62
=	23		63
sto	24		64
1	25		65
)	26		66
=	27		67
sto	28		68
0	29		69
rcl	30		70
1	31		71
÷	32		72
3	33		73
3	34		74
3	35		75
4	36		76
9	37		77
=	38		78
stop	39		79

RANDOM NUMBERS FROM N(0,1).

KEY	#	KEY	#
HALT	00	9	40
rcl	01	=	41
0	02	sto	42
×	03	2	43
5	04	)	44
7	05	=	45
4	06	sto	46
1	07	0	47
−	08	rcl	48
(	09	2	49
+	10	÷	50
1	11	2	51
./EE	12	3	52
./EE	13	./EE	53
9	14	7	54
−	15	1	55
1	16	9	56
./EE	17	4	57
./EE	18	4	58
9	19	4	59
=	20	+	60
sto	21	0	61
1	22	=	62
)	23	cos	63
×	24	×	64
8	25	(	65
5	26	rcl	66
3	27	1	67
7	28	÷	68
−	29	5	69
(	30	7	70
+	31	4	71
1	32	3	72
./EE	33	=	73
./EE	34	x^2	74
9	35	1/x	75
−	36	ln	76
1	37	$\sqrt{x}$	77
./EE	38	)	78
./EE	39	=	79

Normal distribution
mean 0, s.d., 1

Execution:

Store any number between 0 and 1 in memory 0.
run/random number/
run/new random number/
. . . etc.

RANDOM NUMBER GENERATOR

This program generates random numbers that are uniformly distributed on the interval (0, 1). It uses the linear congruential method based on the formula.

$x_{n+1} = (ax_n + c)$ modulo m.

Given a starting value x_0 between 0 and m it generates the sequence of numbers

$r_1 = x_1/m$, $r_2 = x_2/m$,
$r_3 = x_3/m$ . . .

The program is written so as not to require the excessive time taken by some random number generation programs.

To avoid loss of significant digits with this program, a, m and c should be chosen so that

$a \times m + c < 10^{11}$

Suggested values are

a = 24298
c = 9991
m = 199017

Execution:

a/run/c/run/m/run/x_0/run/r_1/
run/r_2/run/ . . .

KEY	#	KEY	#
HALT	00	4	40
sto	01	—	41
1	02	(	42
1	03	rcl	43
./EE	04	3	44
./EE	05	×	45
1	06	rcl	46
1	07	0	47
sto	08	)	48
0	09	=	49
stop	10	gin	50
sto	11	2	51
2	12	6	52
stop	13	sto	53
sto	14	4	54
3	15	goto	55
stop	16	3	56
×	17	9	57
rcl	18	rcl	58
1	19	4	59
+	20	÷	60
rcl	21	rcl	61
2	22	3	62
=	23	=	63
sto	24	stop	64
4	25	rcl	65
rcl	26	4	66
0	27	goto	67
÷	28	1	68
1	29	7	69
0	30		70
—	31		71
sto	32		72
0	33		73
1	34		74
=	35		75
gin	36		76
5	37		77
8	38		78
rcl	39		79

	00		40
	01		41
	02		42
	03		43
	04		44
	05		45
	06		46
	07		47
	08		48
	09		49
	10		50
	11		51
	12		52
	13		53
	14		54
	15		55
	16		56
	17		57
	18		58
	19		59
	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

	00		40
	01		41
	02		42
	03		43
	04		44
	05		45
	06		46
	07		47
	08		48
	09		49
	10		50
	11		51
	12		52
	13		53
	14		54
	15		55
	16		56
	17		57
	18		58
	19		59
	20		60
	21		61
	22		62
	23		63
	24		64
	25		65
	26		66
	27		67
	28		68
	29		69
	30		70
	31		71
	32		72
	33		73
	34		74
	35		75
	36		76
	37		77
	38		78
	39		79

Sinclair Radionics Inc.
Galleria
115 East 57th Street
New York NY 10022
USA

Sinclair Radionics Ltd
London Road
St Ives Huntingdon
Cambridgeshire PE17 4HJ

48584 621

sinclair