```
MLE  <statements> END
```

Statements

Assignment Statement Formats:

```
<var> = <expr>                          {simple assignment}
<var>:<type>                            {simple declaration}
<var>:<type> = <expr>                   {simple declaration with initialization}
<var>:<type>[i TO j]                    {array declaration}
<var>:<type>[i TO j] = <expr>           {array declaration with initialization}
<var>:<type>[i TO j, ...]               {multidimensional array declaration}
<var>:<type>[i TO j, ...] = <expr>      { ... with initialization}
<var>:<type>[i TO j] = [<expr> [<expr>...]]  { ... with data initialization}
<var>:<type>[i TO j, ...] = [[<expr> [<expr>...]], [...] ...]
```

```
BEGIN <statements> END
```

```
BREAK
```

```
CONTINUE
```

```
CURVE [KEY <string> | WITH <string> | AXES <string>]
  <ivar> = <iexpr> TO <iexpr>   |  <rvar> (<exp>, <exp> [<iexp>])
  <expr> <expr> [<expr> ...] [<string> ...]
END
```

```
EXIT
```

```
DATA
  <var> [FIELD i [LINE j]] [= <expr>] [DROPIF <expr> | KEEPIF <expr> ...]
  ...
END
```

```
FOR <var> = <expr> TO <expr> DO <statements> END
FOR <var> = <expr> TO <expr> STEP <expr> DO <statements> END
FOR <var> = <expr> TO <expr> STEPS <iexpr> DO <statements> END
FOR <var> = <array> DO <statements> END
```

```
IF <bexpr> THEN <statements>
[ELSEIF <bexpr> THEN <statements>...]
[ELSE <statements>]
END
```

```
MODEL
  <expr>
RUN  [ THEN <statements> END ]
  FULL [ THEN <statements> END ]
   | REDUCE <paramname>=<expr> [<paramname>=<expr>...] [ THEN <statements> END ]
  .| WITH ..
END
```

```
MULTIPLOT <statements> END
```

```
PLOT <statements> END
```

Procedures:

```
CHDIR(name)                     CLOSE(f)                        DATAFILE(name)
DATATOARRAY(x1, x2, ...)        DEC(x)                          DUMPSYMBOL(x)
DUMPTABLE                       ERASE(name)                     EXEC(cmd, args)
FLUSH(f)                        GETDATE(y [m [d]])              GETTIME(h [m [s [s100]]])
HALT                            INC(x)                          MKDIR(name)
OPENAPPEND(f, name)             OPENREAD(f, name)               OPENWRITE(f, x2)
OUTFILE(<namestr>)              PLOTFILE(<namestr>)             PRINT(x1, x2,...)
PRINTLN(x1, x2,...)             PTRANSFORM(v1, v2, expr)        READ([f,] x1, x2,...)
READLN([f,] x1, x2,...)         RENAME(name1, name2)            RMDIR(name)
SEED(x)                         WRITE([f,] x1, x2,...)          WRITELN([f,] x1, x2,...)
WRITEPLOTLN(x1, x2, ...)        WRITEPLOT(x1, x2, ...)
```

```
REPEAT <statements> UNTIL <bexpr>
```

```
WHILE <bexpr> DO <statements> END
```

Functions

```
DATA [FORM = SUMLL | SUM[MATION] | PROD[UCT]] <expr> END
```

```
DERIVATIVE [(<expr>)] <var> = <expr> [<expr> [<expr>]] END
```

```
FINDMIN <var> (<expr> <expr> [<expr> [<expr> [<expr>]]]) <expr> END
```

```
FINDZERO <var> (<expr> <expr> [<expr> [<expr> [<expr>]]]) <expr> END
```

```
IF <bexpr> THEN <expr> [ELSEIF <bexpr> THEN <expr>...] ELSE <expr> END

INTEGRATE <var> (<expr> <expr> [<expr>]) <expr> END

LEVEL <bexpr> THEN [FORM = SUMLL | SUM[MATION] | PROD[UCT]] <expr> END

LEVELDELTA <expr> THEN [FORM = SUMLL | SUM[MATION] | PROD[UCT]] <expr> END

PARAM <var> [HIGH=<expr>].[LOW=<expr>] [START=<expr>] [TEST=<expr>]
            [FORM=<paramform>]
  [COVAR <expr> <expr> . . . ]
END

PDF <PDF name> (<expr> <expr> ... )
  <expr> <expr> ...
  [HAZARD COVAR <var> <expr> [COVAR <var> <expr> ...]]
END

POSTASSIGN <expr> <statement> END

PREASSIGN <statement> <expr> end

PRODUCT <var> (<expr> <expr> [<expr>]) <exp> END

QUANTILE <PDF name> (<expr> [<expr> [<expr>]])
  <expr> <expr> ...
  [HAZARD COVAR <var> <expr> [COVAR <var> <expr> ...]]
END

SUMMATION <var> (<expr> <expr> [<expr>]) <exp> END
```

Simple Functions

ABS(x)
ADD(x, y)
ANDF(x, y)
ARCCOS(x)
ARCCOSH(x)
ARCCOT(x)
ARCCOTH(x)
ARCCSC(x)
ARCCSCH(x)
ARCSEC(x)
ARCSECH(x)
ARCSIN(x)
ARCSINH(x)
ARCTAN(x)
ARCTANH(x)
ARG(x)
ARGCOUNT
ARGSTRING(x)
BESSELI(x, y)
BESSELJ(x, y)
BESSELK(x, y)
BESSELY(x, y)
BETA(ν, ω)
BOOL2STR(x)
CEIL(x)
CHR(x)
COMB(x, y)
COMP(x)
COMPN(x, n)
CONCAT(x1, x2)
COS(x)
COSH(x)
COT(x)
COTH(x)
CSCH(x)
DEC(x)
DEFALULTOUTNAME
DEFAULTPLOTNAME
DELTA(x, y)
DIVIDE(x, y)
DMSTOD(x, y, z)
DMSTOR(x, y, z)
DMYTOJ(x, y, z)
DTOR(x)
EARTHDIST(x1 x2 x3 x4)
ENVCOUNT
ENVSTRING
EOF(f)
EOLN(f)
ERF(x)
ERFC(x)
EXEC(s)
EXISTS(s)
EXP(x)
FACT(x)
FILESIZE(x)
FISHER(x)
FISHERINV(x)
FLOOR(x)
FRAC(x)
GAMMA(x)
GCF(x, y)
GETDIR
GETENV(x)
HEAVISIDE(x)
IBETA(p, ν, ω)
IBETAC((p, ν, ω)
IDIV(x, y)
IGAMMA(x, y)
IGAMMAC(x1, x2)
IGAMMAE(x1, x2)
IM(x)
INC(x)
INT(x)
INT2STR(x)
INVCHISQ(p, d)
INVERT(x)
INVNORMAL(p)
IRAND(x, y)
ISANINFINITY(x)
ISEQ(x, y)
ISEVEN(x)
ISGE(x, y)
ISGT(x, y)
ISINFINITY(x)
ISLE(x, y)
ISLT(x, y)
ISNAN(x)
ISNE(x, y)
ISNEAR(x, b, δ)
ISNEGINFINITY(x)
ISODD(x)
JULIAND(x)
JULIANM(x)
JULIANY(x)
LCM(x, y)
LEAPYEAR(y)
LEFTSTRING(x, y)
LN(x)
LNFACT(x)
LNGAMMA(x)
LOG(x)
LOG10(x)
LOGBASE(x, y)
LOGISTIC(x)
LOGIT(x)
LUNARPHASE(j)
MAX(x, y)
MIN(x, y)
MIX(p, x, y)
MODULO(x, y)
MONTHDAYS(m, y)
MULTIPLY(x, y)
NEGATE(x)
NOTF(x)
ORD(c)
ORF(x, y)
PERMUTATIONS(x, y)
POLARTORECTX(r, a)
POLARTORECTY(r, a)
POWER(x, y)
PUT(x)
RAND
RE(x)
REAL2STR(x, l, s)
RECTTOPOLARA(x, y)
RECTTOPOLARR(x, y)
RECTTOSPHEREA1(x, y, z)
RECTTOSPHEREA2(x, y, z)
RECTTOSPHERER(x, y, z)
REMAINDER(x, y)
RIGHTSTRING(x, y)
ROOT(x, y)
ROUND(x)
RRAND(x, y)
RTOD(x)
SEC(x)
SECH(x)
SETRANSFORM(expr)
SGN(x)
SHIFTLEFT(x, y)
SHIFTRIGHT(x, y)
SIGN(x, y)
SIN(x)
SINH(x)
SPHERETORECTX(r, a1, a2)
SPHERETORECTY(r, a1, a2)
SPHERETORECTZ(r, a1, a2)
SQR(x)
SQRT(x)
STANDARDIZE(x, μ, σ)
STRING2INT(s)
STRING2REAL(s)
STRINGLEN(x)
SUBSTRING(x, y, z)
SUBTRACT(x, y)
TAN(x)
TANH(x)
TOBASE(x)
TOLOWER(x)
TOUPPER(x)
TRIM(x)
TRIML(x)
TRIMR(x)
TRUNC(x)
WEEKDAY(x)
XORF(x, y)
YEARDAY(x)
ZETA(x)

Param Forms

ADD
DIVIDE
EXCESS
EXPADD
FISHER
FISHERINV
INVADD
INVERT
INVLOGLIN
INVMULTIPLY
LOGISTIC
LOGIT
LOGLIN
MULTIPLY
NUMBER
POWER
POWEREXP

PDF Names

ALPHA
ARCSINE
ASYMPTOTICRANGE
BERNOULLITRIAL
BETA
BINOMIAL
BIRNBAUMSAUNDERS
BIVNORMAL
CAUCHY
CHI
CHISQUARED
COMPOUNDEXTREME
DANIELS
DISK
EXPONENTIAL
FAILED
FOLDEDNORMAL
GAMMA
GAMMAFRAIL
GAUSSIAN
GENGAMMA
GENGUMBEL
GEOMETRIC
GOMPERTZ
GUMBEL
HORSESHOE
HYPERBOLICSECANT
HYPERGEOMETRIC
HYPER2EXP
HYPO2EXP
IMMUNE
INVBETA1
INVBETA2
INVCHI
INVGAMMA
INVGAUSSIAN
LAPLACE
LARGEEXTREME1
LARGEEXTREME2
LINEARHAZARD
LNGAMMA
LNLOGISTIC
LNNORMAL
LOGISTIC
LOGNORMAL
LOGSERIES
LOWMAX
MAKEHAM
MAXWELL
MIXMAKEHAM
NEGBINOMIAL
NEGHYPERGEOMETRIC
NORMAL
PARETO
PASCAL
POISSON
POLYAEGGENBERGER
POWERFUNCTION
RAISEDCOSINE
RANDOMWALK
RAYLEIGH
RECTANGULAR
REVPOWERFUNCTION
RINGINGEXP0
RINGINGEXP180
SHIFTEXPONENTIAL
SHIFTGAMMA
SHIFTLOGNORMAL
SHIFTWEIBULL
SILER
SMALLEXTREME1
SMALLEXTREME2
STERILE
SUBBOTIN
THOMAS
UNIFORM
VONMISES
WEIBULL
ZIPF

Predefined Variables and Constants

AIC_SELECT
AICC_SELECT
ALT_LOGISTIC
ANNEALING
ATOMICMASSU
AVOGADROSN
BATCHMODE
BIC_SELECT
BOHRMAGNETON
BOHRRADIUS
BOLTZMANNSC
BRENT_ITS
BRENT_MAGIC
CGRADIENT1
CGRADIENT2
CI_CHISQ
CI_CONVERGE
CI_EVALS
CI_LIMIT_DELTA
CI_MAXITS
COMPLEXDECIMALS
COMPLEXWIDTH
CONVERGE_FLAG
CREATE_OBS
D_IDX
DATADELIMITERS
DATAFILENAME
DEBUG
DEBUG_DATA
DEBUG_ECHO
DEBUG_EXEC
DEBUG_INT
DEBUG_LIK
DEBUG_PARSE
DEBUG_SYM
DEGREESPERRADIAN
DELTA_LL
DIFF_DX
DIRECT
DIST_DX_SCALE
DIST_T_END
DIST_T_N
DIST_T_START
DROPPED_OBS
DX_MAXITS
DX_START
DX_TOOBIG
DX_TOOSMALL
E
EPSILON
EULERSC
EVALS
EXP_HAZARD
FALSE
FDATA
FIND_EPS
FIND_MAXITER
FINPUT
FOUTFILE
FOUTPUT
FPLOT
FPLOTDATA
FREE_PARAMS
GNUPLOT
GNUPLOTINIT
GRAVITATIONALC
HIGH_DEFAULT
INFINITY
INFO_METHOD1
INFO_METHOD2
INPUT_SKIP
INTEGRATE_METHOD
INTEGRATE_N
INTEGRATE_TOL
INTERMPLOT
INVERT_FLAG
ITERATIONS
ITERATION_PRINT
I_AQUAD
I_SIMPSON
I_TRAP_CLOSED
I_TRAP_OPEN
LARGEST_LIKELIHOOD
LARGEST_LLIKELIHOOD
LARGE_ZERO
LIGHTC
LINE_NUMB
LNINFINITY
LOGLIKELIHOOD
LOG_10
LOW_DEFAULT
MACHINE_EPSILON
MAXEVALS
MAXINT
MAXITER
MAX_BOOLEANS
MAX_CHARS
MAX_INTEGERS
MAX_REALS
MAX_STRINGS
METHOD
METHOD_LOOP
MINIMUM_ITS
MIN_SIGNIFICANT
MLEBASEFILE
MPLOTXSCALE
MPLOTYSCALE
MULTIPLOTINIT
NAN
NEGINFINITY
NEWTON
N_OBS
N_VARS
oo
OSSEP
OSVERSION
OUTFILENAME
PARSE_ONLY
PI
PLANCKINV2PI
PLANCKSC
PLOT_DISTS
PLOTFILENAME
PLOTINIT
PLOTPOINTS
POWELL
PRINT_BASIC
PRINT_CI
PRINT_COUNTS
PRINT_DATA_STATS
PRINT_DISTS
PRINT_FIELDS
PRINT_FREE_PARAMS
PRINT_INFO
PRINT_LLIKS
PRINT_OBS
PRINT_PARAMS
PRINT_REDUCED
PRINT_SE
PRINT_SHORT
PRINT_VCV
PROGRAM_NAME
RADIANSPERDEGREE
RANDOM_SEED
READDELIMITERS
READSTARTFILE
REALDECIMALS
REALWIDTH
RELEASE
REVISION
RYDBERGC
SA_ADJ_CYCLES
SA_ADJ_LOWERBOUND
SA_ALT_ADJUSTMENT
SA_COOLING
SA_EPS_NUMBER
SA_STEPLENGTH
SA_STEPLENGTH_ADJ
SA_STEPS
SA_TEMPERATURE
SECONDSPERDAY
SIMPLEX
SIMPLEX_ALPHA
SIMPLEX_BETA
SIMPLEX_GAMMA
SMALLEST_LIKELIHOOD
SMALLEST_LLIKELIHOOD
SMALLEST_NUMBER
SQRT_EPSILON
START_DEFAULT
SURFACE_POINTS
SYMBOLICINFIN
SYSTEM
TERMFILE
TEST_DEFAULT
TITLE
TOTAL_OBS
TRUE
UNIVERSALGASC
VCV_EVALS
VCV_WIDTH
VERBOSE
VERSION
WRITEDELIMITER
WRITESTARTFILE

Number Formats

Format	Examples	Conversion	Result
D	1, 200		integer
d.d, *d.*	3.1415, 3.		real
ds, *-ds*, *d.ds*, *-d.ds*,	14%, 23.7M, 45.7da, 2n, 2.418E	Suffix (see below)	real
*d*E*d*, *d*E-*d*, *d.d*E*d*, *d.d*E-*d*, *d*.E*d*, *d*.E-*d*	3e23, 511E-10, 31.416e-1, 7.0E-10, 12.e-6, 1.45E-3, 1.0E0	Standard exponential format. $xEy \Rightarrow x \times 10^y$	real
0R*v*	0RXLVII, 0rMXVI, 0rmdclxvi	Roman numerals to integer	integer
*d*X*y*	2x1001 (binary), 8X3270 (octal), 16xA4CC (hex), 32x3vq4h (base 32).	Converts *y* from base *d* (from 2 to 36) into integer.	integer
d:*d*:*d*, *d*:*d*:*d.d*, *d*:*d*, *d*:*d.d*	10:42, 14:55:32, 10:40:23.4, 16:53.2	24-hour time into hours. Hours must be 0-24.	real
d:*d*:*d*AM, *d*:*d*:*d*PM, *d*:*d*:*d.d*AM, *d*:*d*:*d.d*PM, *d*:*d*PM, *d*:*d*AM, *d*:*d.d*AM, *d*:*d.d*PM	10:42AM, 2:55:32pm, 10:40:23.4am	12-hour time with AM and PM suffixes into hours. Hours must be 0-12.	real
*d*Hd'*d*", *d*Hd'*d.d*", *d*Hd', *d*H*d.d*", *d*H*d.d*"	230h16'32", 14H32'6", 100h22', 30H32.2', 0h12', 0H12'3"	Degree/hour minute, second format. Converted to real angle/time.	real
d`*d*'*d*", *d*`*d*'*d.d*", *d*`*d*', *d*`*d.d*", *-d*`*d*'*d*", *d*`, *d.d*`, *d*°*d*'*d*", *d*°*d*'*d.d*", *d*°*d*', *d*°*d.d*", *d*°, *d.d*°	230`16'32", 14`32'6", 100`22', 30`32.2', 14`, 230°16'32", 14°32'6", 270°10'0", 30°18.2', 3.4°	Degree, minute, second format, converted to radians.	real
d'*d*", *d*'*d.d*", *d*', *d.d*', *d*", *d.d*"	12'32", 166'12.9", 19', 14.7', 12", 607.3"	Minute-second and second format, converted to radians.	real
*d*_*d*/*d*	12_5/16, 3_2/3, 0_1/7	Fraction notation.	real
*d*D*d*M*d*Y	16d12m1944y, 1D6M1800Y	Date converted to Julian day	integer
*d*M*d*D*d*Y	12m16d1944y, 6M1D1800Y	Date converted to Julian day	integer
*d*Y*d*M*d*D	1944y12m16d, 1800Y6M1D	Date converted to Julian day	integer
*d*mmm*y*	14Dec1999, 30jun1961, 1MAY1944	Date converted to Julian day	integer

d is a strings of one or more positive digits; *s* is a one or two character case-sensitive metric or percent suffix , *v* is a string of one or more Roman numeral digits {IVXLCDM}, *y* is a string of one or more characters, *mmm* is a 3-character English month name (jan, Feb, MAR, etc). The degree (°) and micro (µ) characters are available on some hardware platforms as ASCII codes 230 and 248 respectively. On many Intel platforms, the characters are availabel by holding down the <ALT> key and type the code on the numeric keypad.

Metric and Other Number Suffixes

Suffix	Name	Conversion	Suffix	Name	Conversion	Suffix	Name	Conversion
da	deka	$\times 10$	d	deci	$\times 10^{-1}$	Ki	kibi	$\times 2^{10}$
h	hecto	$\times 10^{2}$	c, %	centi, percent	$\times 10^{-2}$	Mi	mebi	$\times 2^{20}$
k	kilo	$\times 10^{3}$	m	milli	$\times 10^{-3}$	Gi	gibi	$\times 2^{30}$
M	mega	$\times 10^{6}$	µ, u	micro	$\times 10^{-6}$	Ti	tebi	$\times 2^{40}$
G	giga	$\times 10^{9}$	n	nano	$\times 10^{-9}$	Pi	pebi	$\times 2^{50}$
T	tera	$\times 10^{12}$	p	pico	$\times 10^{-12}$	Ei	exbi	$\times 2^{60}$
P	peta	$\times 10^{15}$	f	femto	$\times 10^{-15}$			
E	exa	$\times 10^{18}$	a	atto	$\times 10^{-18}$			
Z	zetta	$\times 10^{21}$	z	zepto	$\times 10^{-21}$			
Y	yotta	$\times 10^{24}$	y	yocto	$\times 10^{-24}$			