Mathematical System Functions

The following mathematical functions are supported in arithmetic processing statements (`ADD`, `COMPUTE`, `DIVIDE`, `MULTIPLY`, `SUBTRACT`) and in logical condition criteria:

Function	Format/Length	Explanation
`ABS`*(field)*	same as *field*	Absolute value of *field*.
`ATN`*(field)*	F8 (*)	Arc tangent of *field*.
`COS`*(field)*	F8 (*)	Cosine of *field*.
`EXP`*(field)*	F8 (*)	Exponentiation of exponent *field* to base e , that is, e^{field} , where e is Euler's number.
`FRAC`*(field)*	same as *field*	Fractional part of *field*.
`INT`*(field)*	same as *field*	Integer part of *field*.
`LOG`*(field)*	F8 (*)	Natural logarithm of *field*.
`SGN`*(field)*	same as *field*	Sign of *field* (-1, 0, +1).
`SIN`*(field)*	F8 (*)	Sine of *field* .
`SQRT`*(field)*	F8 (*)	Square root of *field*. A negative value in the argument field will be treated as positive.
`TAN`*(field)*	F8 (*)	Tangent of *field*.
`VAL`*(field)*	same as target field	Extract numeric value from an alphanumeric *field*. The content of the *field* must be the alphanumeric (code page or Unicode) character representation of a numeric value. Leading or trailing blanks in the *field* will be ignored; decimal point and leading sign character will be processed. If the target field is not long enough, decimal digits will be truncated (see also *Field Truncation and Field Rounding* in the section *Rules for Arithmetic Assignment* of the *Programming Guide*).

[*] These functions are evaluated as follows:

- The argument is converted to format/length F8 and then passed to the operating system for computation.

- The result returned by the operating system has format/length F8, which is then converted to the target format.

A *field* to be used with a mathematical function - except `VAL` - may be a constant or a scalar; its format must be numeric (N), packed numeric (P), integer (I), or floating point (F).

A *field* to be used with the VAL function may be a constant, a scalar, or an array; its format must be alphanumeric.

Mathematical Functions Example:

```
** Example 'MATHEX': Mathematical functions
************************************************************************
DEFINE DATA LOCAL
1 #A     (N2.1) INIT <10>
1 #B     (N2.1) INIT <-6.3>
1 #C     (N2.1) INIT <0>
1 #LOGA  (N2.6)
1 #SQRTA (N2.6)
1 #TANA  (N2.6)
1 #ABS   (N2.1)
1 #FRAC  (N2.1)
1 #INT   (N2.1)
1 #SGN   (N1)
END-DEFINE
*
COMPUTE #LOGA = LOG(#A)
WRITE NOTITLE '=' #A 5X 'LOG'         40T #LOGA
*
COMPUTE #SQRTA = SQRT(#A)
WRITE        '=' #A 5X 'SQUARE ROOT' 40T #SQRTA
*
COMPUTE #TANA  = TAN(#A)
WRITE        '=' #A 5X 'TANGENT'     40T #TANA
*
COMPUTE #ABS   = ABS(#B)
WRITE     //  '=' #B 5X 'ABSOLUTE'    40T #ABS
*
COMPUTE #FRAC  = FRAC(#B)
WRITE        '=' #B 5X 'FRACTIONAL'  40T #FRAC
*
COMPUTE #INT   = INT(#B)
WRITE        '=' #B 5X 'INTEGER'     40T #INT
*
COMPUTE #SGN   = SGN(#A)
WRITE     //  '=' #A 5X 'SIGN'        40T #SGN
*
COMPUTE #SGN   = SGN(#B)
WRITE        '=' #B 5X 'SIGN'        40T #SGN
*
COMPUTE #SGN   = SGN(#C)
WRITE        '=' #C 5X 'SIGN'        40T #SGN
*
END
```

Output of program MATHEX:

```
#A:  10.0      LOG                     2.302585
#A:  10.0      SQUARE ROOT             3.162277
#A:  10.0      TANGENT                 0.648360


#B:  -6.3      ABSOLUTE                 6.3
#B:  -6.3      FRACTIONAL              -0.3
#B:  -6.3      INTEGER                 -6.0
```

```
#A:  10.0    SIGN                      1
#B:  -6.3    SIGN                     -1
#C:   0.0    SIGN                      0
```