© INNOVATIVE RESEARCH THOUGHTS | Refereed | Peer Reviewed | Indexed
ISSN : 2454 – 308X | Volume : 03 , Issue : 08 | October – December 2017

Study of various Arithmetic Expressions in Fortran

Sunita

Abstract : Arithmetic expressions are extremely important in fundamental computer syntax because they provide numeric values that support code functions. By contrast, other kinds of expressions, such as character expressions or Boolean expressions, contain different kinds of indicators.

Character expressions contained text values or individual letters or characters for analysis or display, while Boolean expressions contain one of two Boolean values: true or false.

There are two different kinds of arithmetic expressions in computer programming syntax: integers or real numbers, and real or floating point numbers. The latter are used to identify and store complex numbers that may not fit into an integer value.

In computer code, operators and functions work on individual expressions or sets of expressions including arithmetic, character and Boolean expressions. These provide the basis for the kinds of data work that go on within a software program.

Assignment Statements

The Assignment Statement (=) is used to assign values to variables. The variable in this case can be thought of as a storage location in the computer.

Assignment has the general form:

variable = expression

where expression can be a constant, another variable or a formula (math expression).

Important: assignment does not denote equality, but rather, replacement. In this sense, maybe it should look like:

variable <-- expression

Example: N = N + 1 is not a correct mathematical statement, but is valid in FORTRAN.

There should be a single variable name on the left-hand side since assignment is to a variable or storage location.

INVALID: 5 = N

A + 10.3 = LENGTH

A = B = C

Arithmetic Operations

FORTRAN variables and constants can be processed using operations and functions appropriate to their types.

The five arithmetic operators in FORTRAN are:

1. Addition +
2. Subtraction -
3. Multiplication *
4. Division /
5. Exponentiation **

Parentheses can modify the order of evaluation. Expressions within parentheses will be evaluated first, using the priority rules. If parentheses are nested, the inner expression is evaluated first. Using parentheses is recommended (even if their use does not change the order of evaluation) because it makes the expression easier to read.

Type Conversion and Mixed-Mode Arithmetic

FORTRAN allows you to mix numeric data of different type together in arithmetic expressions. Type conversion occurs when you mix variables or constants of different types together in an expression. The result is the higher of the two operands in the following hierarchy:

DOUBLE PRECISION (NOTE: numeric data types ONLY)

REAL

INTEGER

Example: INTEGER * REAL ---> REAL

INTEGER / REAL ---> REAL

INTEGER + OR - REAL ---> REAL

REAL * DOUBLE ---> DOUBLE PRECISION

10 * 2 --> 20

10 * 2.0 --> 20.0

If you assign a data item to a variable of a different numeric type, it is converted to that type

Example: INTEGER I

I = 10.5 (10 is stored)

X = I (10.0 is stored)

REAL ALPHA

ALPHA = 20 (20.0 is stored)

Beware of integer division. Dividing one integer by another causes truncation of any decimal part.

Example: 1.0 / 2.0 = 0.5 but 1 / 2 = 0

1. / 2 = 0.5 and 1 / 2.= 0.5

Built-In FORTRAN Library Functions

FORTRAN provides a library of pre-defined functions which you can use in your programs, but not nearly the number available in Matlab.

Use functions as you would in an equation. You can use constants, variables, and even expressions as the argument. Parentheses are used to enclose the argument.

Example: COS(ANGLE) EXP(A + B) SQRT(124)

X = ABS(A+B+C)

Manipulating Character Data

Character data, being non-numeric, cannot be "added", "subtracted", "multiplied", etc... as numeric data can. However, FORTRAN does provide ways of manipulating CHARACTER data. The concatenation operation can be used to combine character constants and variables. Concatenation is denoted by the operator //. For example:

'SO'//' WHAT' --> 'SO WHAT'

STR1 = 'CONCAT'

STR2 = STR1//'ENATION' --> 'CONCATENATION'

Another operation commonly performed on character strings is extracting substrings from another string variable. A substring is extracted from another string by using the first and last character positions separated by a colon (:). Using the above example:

STR2(4:6) --> 'CAT'

STR2(4:4+2) --> 'CAT'

Character string variables have defined lengths. This can introduce two problems when assigning constants to string variables if the length is not the same:

* If the constant is LONGER than the variable's defined length, the constant will be truncated on the right.

* If the constant is SHORTER than the variable's defined length, blanks will be added (padded) on the right.

Example: CHARACTER DAY*4

DAY = 'MONDAY' ('MOND' stored)

DAY = '12' ('12 ' stored)

References :

1. https://www.tutorialspoint.com/fortran/fortran_constants.htm
2. http://www.mathcs.emory.edu/~cheung/Courses/561/Syllabus/5-Fortran/variables.html
3. http://www.infis.univ.trieste.it/fortran/constant.html
4. http://www.oc.nps.edu/~bird/oc3030_online/fortran/basics/basics.html