Subject: Re: IDL's handling of LOGICAL quantities (WHERE)
Posted by Pavel Romashkin on Tue, 12 Oct 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Maybe NOT function in IDL is not exactly what some people are used to, but at least it is documented. NOT function performs a biwise conversion, and an expected result will only come if the tested value is truly boolean. Here is what Help says:
"NOT
The NOT operator is the Boolean inverse and is a unary operator (it has only one operand). In other words, "NOT true" is equal to "false" and "NOT false" is equal to "true." NOT complements each bit for integer operands.
Note Signed integers are expressed using the "2s complement" representation. This means that to arrive at the decimal representation of a negative binary number (a string of binary digits with a one as the most significant bit), you must take the complement of each bit, add one, convert to decimal, and prepend a negative sign. This means that NOT 0 equals -1, NOT 1 equals -2, etc.
For floating-point operands, the result is 1.0 if the operand is zero; otherwise, the result is zero. The NOT operator is not valid for string or complex operands.
NOT 5 = -6

NOT 0101 = 1010"

It is easy tget used to it, especially because other methods of comparison are readily available. It is common in IDL that logical functions return integer values - we better get used to it.
Good luck,
Pavel

James Tappin wrote:

> \begin{rant}
> I've finally decided to have a public moan about one of the weaknesses of IDL's
> handling of logical operations: to boot -- that the WHERE function follows
> a C-like interpretation while most other things are Fortran-like.
>
> for example suppose we have an array (m) some of whose values are NaN then the
> (inefficient) loop:
> for j=0, n_elements(m) do if not finite(m(j)) then m(j)=0
> will set all non-finite elements of m to 0.
> However:
> m(where(not finite(m))) = 0
> will zero out the whole array since where sees (not 1) as a Yes.
> [The correct solution is of course:
> m(where(finite(m) ne 1)) = 0
>]
>
> Or a simpler example:
> IDL> a = [0, 1, 0, 1]

> IDL> print, where(a eq 0)
> 0 2
> IDL> print, where(not (a ne 0))
> 0 1 2 3
>
> I guess the proper answer isto have aproper logical or boolean type and
> functions like FINITE and logical operations should return it, and of course
> WHERE should accept it.
