Subject: Re: Bug in operator precedence
Posted by Foldy Lajos on Tue, 16 Aug 2005 08:43:12 GMT
View Forum Message <> Reply to Message

Hi,

print, ~1+1, (~1)+1, ~(1+1) gives 1 1 0, which shows, that ~ has precedence greater or equal to addition. According to the documentation, this is clearly wrong (and GDL is right printing 0 1 0).

By the way, in GDL print, NOT 1+1, (NOT 1)+1, NOT (1+1) gives -3 -1 -3, which should be -1 -1 -3, because the precedence of NOT is equal to addition (and IDL is right).

Now, we have two similar bugs, and the efforts needed to fix them are equal. Which bug will be fixed first? (I would bet on GDL's bug fixed in a few days.)

Ready, Steady, Go! :-))))))))

regards,
lajos

On Tue, 16 Aug 2005, m_schellens@hotmail.com wrote:

```
>  According to the manual, operators
>
>  ~ || &&
>
>  have lower precedence than
>
>  AND OR XOR
>
>  Now I get:
>  IDL>  print, 1 && ~3 and 4
>     0
>  IDL>  print, 1 && (~3) and 4
>     0
>  IDL>  print, 1 && ~(3 and 4)
>     1
>
>  I would consider this as a bug.
>  Anybody agree or am I missing something?
>
>  Cheers,
>  marc
```

>
>

Subject: Re: Bug in operator precedence
Posted by marc schellens[1] on Tue, 16 Aug 2005 08:56:50 GMT
View Forum Message <> Reply to Message

It is already fixed :-)
(in the CVS)
also the ~ operator has now the same precedence as NOT like in IDL

Cheers,
marc

Subject: Re: Bug in operator precedence
Posted by Mark Hadfield on Tue, 16 Aug 2005 22:20:07 GMT
View Forum Message <> Reply to Message

m_schellens@hotmail.com wrote:
> According to the manual, operators
>
> ~ || &&
>
> have lower precedence than
>
> AND OR XOR
>
> Now I get:
> IDL> print, 1 && ~3 and 4
>   0
> IDL> print, 1 && (~3) and 4
>   0
> IDL> print, 1 && ~(3 and 4)
>   1
>
> I would consider this as a bug.

Yes, but I suggest that the best way to fix it is to change the manual.
I think "~" logical negation should have similar priority to "not"
(bitwise negation) and higher than any of the binary logical operators,
ie the "eq"s the "and"s and the "&&"s. Why do I think this? Well...

  * Bitwise and logical negation are very similar conceptually
  * Unary operators normally outrank binary ones
  * Like many people, I don't bother to read the manual unless I

really need to, and I have code that relies on what IDL does,
not what the manual says it should do.

--
Mark Hadfield "Kei puwaha te tai nei, Hoea tahi tatou"
m.hadfield@niwa.co.nz
National Institute for Water and Atmospheric Research (NIWA)

Subject: Re: Bug in operator precedence
Posted by Peter Mason on Tue, 16 Aug 2005 22:47:13 GMT
View Forum Message <> Reply to Message

Mark Hadfield wrote:
<...>
> * Like many people, I don't bother to read the manual unless I
> really need to, and I have code that relies on what IDL does,
> not what the manual says it should do.

The other day, a colleague pointed out to me that if you read a manual it
means that you have failed :-)

Subject: Re: Bug in operator precedence
Posted by marc schellens[1] on Wed, 17 Aug 2005 03:18:01 GMT
View Forum Message <> Reply to Message

Maybe the last statement was ambiguous, should be:
...
also the ~ (logical negation) operator has now the same precedence as
the NOT (bitwise negation) operator, like in IDL
...

Subject: Re: Bug in operator precedence
Posted by Foldy Lajos on Wed, 17 Aug 2005 07:05:49 GMT
View Forum Message <> Reply to Message

hi,

RSI does the same, it has constructs like

```
  if (measure eq 3 && ~N_ELEMENTS(powerIn)) then ...
```

in his library, which rely on ~ having higher precedence than &&.

regards,
lajos

On Wed, 17 Aug 2005, Mark Hadfield wrote:

> * Like many people, I don't bother to read the manual unless I
> really need to, and I have code that relies on what IDL does,
> not what the manual says it should do.