
Subject: Re: array subscript conversion

Posted by [Dick Jackson](#) on Wed, 25 Jul 2007 17:46:49 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hi,

news:Pine.LNX.4.64.0707251721430.28405@bifur.rmki.kfki.hu...

> Hi guys,

>

> according to the manual, array subscripts are converted to long (or long64 on
> 64 bit systems) before use if necessary, so an explicit conversion should not
> affect the result.

>

> IDL> print, !version

> { x86 linux unix linux 6.3 Mar 23 2006 32 64 }

> IDL>

> IDL> a=lindgen(10)

> IDL> print, a[[long(-1ull)]]

> 0

> IDL> print, a[[-1ull]]

> 9

>

> Is it a bug or I am missing something?

I think you're expecting -1ull to be negative, but the 'u' in 'ull' means
'unsigned'. What you end up with instead of -1 is the largest 64-bit integer
(this is a nice shortcut when it's actually what you want to do!):

IDL> help,-1ull

<Expression> ULONG64 = 18446744073709551615

That's why a[[-1ull]] gives 9, as the subscript is (somewhat) larger than the
maximum index in the array a. It is similar to a[[11]] below...

IDL> help,long(-1ull)

<Expression> LONG = -1

This is similar to a[[-1]] below...

IDL> print,a[[-1]]

0

IDL> print,a[[11]]

9

And we know this is different from simple subscripting which doesn't allow
out-of-range values:

```
IDL> print,a[-1]
Attempt to subscript A with <INT    (    -1)> is out of range.
Execution halted at: $MAIN$
IDL> print,a[11]
Attempt to subscript A with <INT    (    11)> is out of range.
Execution halted at: $MAIN$
```

--
Cheers,
-Dick

--
Dick Jackson Software Consulting <http://www.d-jackson.com>
Victoria, BC, Canada +1-250-220-6117 dick@d-jackson.com
