Subject: Re: Oh No...
Posted by Paolo Grigis on Wed, 18 Apr 2007 12:38:38 GMT
View Forum Message <> Reply to Message

mfein2@gmail.com wrote:
> I've just discovered, after many years of using IDL, that expressions
> have a value:
>
> IDL> print, (x = 5)
>   5
>
> The possibilities for Obfuscated IDL have now gone to 11 on a scale
> from 0 to 10.
>

Speaking of which, what about

```
IDL> delvar,a ;undefines a
IDL> a[(a=2*findgen(10))]=1 ;agreed, this is a bit crazy, but hold on
IDL> print,a
      1.00000      1.00000      1.00000      6.00000      8.00000      10.0000      1.00000
      14.0000      1.00000      1.00000
```

Now, this must be a bug, surely... (or a very strange feature indeed).

Ciao,
Paolo

Subject: Re: Oh No...
Posted by Foldy Lajos on Wed, 18 Apr 2007 13:04:59 GMT
View Forum Message <> Reply to Message

On Wed, 18 Apr 2007, Paolo Grigis wrote:

```
>
> Speaking of which, what about
>
> IDL> delvar,a ;undefines a
> IDL> a[(a=2*findgen(10))]=1 ;agreed, this is a bit crazy, but hold on
> IDL> print,a
>      1.00000      1.00000      1.00000      6.00000      8.00000
> 10.0000      1.00000
>      14.0000      1.00000      1.00000
>
> Now, this must be a bug, surely... (or a very strange feature indeed).
>
```

No, invalid code, with undefined result :-)

We have no formal definition of the IDL syntax, but there is a rule in
Fortran/C/C++/... which should be true for IDL, too: a memory cell can
be modified at most once in an expression. You are modifying 'a' twice
in a single expression.

(Section 6.5#2 of the C99 specification: "Between the previous and next
sequence point an object shall have its stored value modified at most once
by the evaluation of an expression. Furthermore, the prior value shall be
accessed only to determine the value to be stored.")

regards,
lajos

Subject: Re: Oh No...
Posted by Paolo Grigis on Wed, 18 Apr 2007 13:45:56 GMT
View Forum Message <> Reply to Message

Fï¿½LDY Lajos wrote:
```
>
>  On Wed, 18 Apr 2007, Paolo Grigis wrote:
>
>>
>>  Speaking of which, what about
>>
>>  IDL> delvar,a ;undefines a
>>  IDL> a[(a=2*findgen(10))]=1 ;agreed, this is a bit crazy, but hold on
>>  IDL> print,a
>>       1.00000      1.00000      1.00000      6.00000      8.00000
>>  10.0000      1.00000
>>       14.0000      1.00000      1.00000
>>
>>  Now, this must be a bug, surely... (or a very strange feature indeed).
>>
>
>  No, invalid code, with undefined result :-)
>
>  We have no formal definition of the IDL syntax, but there is a rule in
>  Fortran/C/C++/... which should be true for IDL, too: a memory cell can
>  be modified at most once in an expression. You are modifying 'a' twice
>  in a single expression.
```

Yes, this seems a sensible precaution, but then I think that in the interest
of safety it may be better if such an expression would throw a compiler
or at least a runtime error in IDL... I don't think there is much in the

way of a sensible usage for such kind of expressions, so not much would
be lost.

On the other hand, a similar example which does not access memory cells
out of the array boundaries seems to function more or less as one would
expect:

```
IDL> a[(a=0.5*findgen(10))]=7*findgen(10)
IDL> print,a
      7.00000      21.0000      35.0000      49.0000      63.0000      2.50000      3.00000
      3.50000      4.00000      4.50000
```

so in IDL such expression seems to be valid, and the innermost array
is generated first, and its values are then used for indexing itself...

Ciao,
Paolo

>
> (Section 6.5#2 of the C99 specification: "Between the previous and next
> sequence point an object shall have its stored value modified at most
> once by the evaluation of an expression. Furthermore, the prior value
> shall be accessed only to determine the value to be stored.")
>
> regards,
> lajos

Subject: Re: Oh No...
Posted by Michael Galloy on Wed, 18 Apr 2007 18:01:53 GMT
View Forum Message <> Reply to Message

On Apr 18, 7:45 am, Paolo Grigis <pgri...@astro.phys.ethz.ch> wrote:
> FÖLDY Lajos wrote:
>
>> No, invalid code, with undefined result :-)

I don't think it invalid or undefined (very confusing, yes). a is
created and used to index itself following the "normal" rules of
indexing in IDL.

Remember that it is valid to use an index array that has indices out
of range, they are simply brought back into range:

```
IDL> b = findgen(10)
IDL> b[[-1, 20]] = 1
IDL> print, b
     1.00000      1.00000      2.00000      3.00000      4.00000
```

```
5.00000      6.00000
      7.00000      8.00000      1.00000
```

You can turn this behavior off with COMPILE_OPT:

```
IDL> compile_opt strictarrsubs
IDL> a[(a=2*findgen(10))]=1
% Array used to subscript array contains out of range subscript: A.
% Execution halted at: $MAIN$
```

Mike
--
www.michaelgalloy.com