Subject: precedence question
Posted by Foldy Lajos on Mon, 14 Mar 2005 18:35:23 GMT
View Forum Message <> Reply to Message

Hi guys,

I am a little bit lost in operator precedence. I think that . and [] has
the same precedence, and associate from left to right. So I expect the
following three expressions to be equivalent:

a=replicate({l:lindgen(2,3)}, 4,5)

help,   a[3:6] .l [*,0:1] ; a
help, (a[3:6]).l [*,0:1] ; b
help, ((a[3:6]).l)[*,0:1] ; c

a is the same as b, but c differs. Where am I wrong?

regards,
lajos

Subject: Re: precedence question
Posted by marc schellens[1] on Tue, 15 Mar 2005 07:53:41 GMT
View Forum Message <> Reply to Message

[] has higher (the highest after (expr)) precedence.

Cheers,
marc

Subject: Re: precedence question
Posted by Foldy Lajos on Tue, 15 Mar 2005 09:42:43 GMT
View Forum Message <> Reply to Message

Hi Marc,

On Tue, 14 Mar 2005, m_schellens@hotmail.com wrote:

> [] has higher (the highest after (expr)) precedence.
>
> Cheers,
> marc
>
>

as concatenation, yes, but not as subscripting. The IDL reference guide says:

Operator Precedence

The following table lists IDL's operator precedence. Operators with the highest precedence are evaluated first. Operators with equal precedence are evaluated from left to right.

Priority	Operator
First (highest)	( ) (parentheses, to group expressions) [ ] (brackets, to concatenate arrays)
Second	. (structure field dereference) [ ] (brackets, to subscript an array) ( ) (parentheses, used in a function call)
Third	* (pointer dereference)

.
.
.

so my original question still holds.

regards,
lajos

Subject: Re: precedence question
Posted by marc schellens[1] on Tue, 15 Mar 2005 12:43:14 GMT
View Forum Message <> Reply to Message

What do you want to hear?
Obviously the docmentation is wrong here and brackets have a higher priority.
marc

Subject: Re: precedence question
Posted by Foldy Lajos on Tue, 15 Mar 2005 13:55:16 GMT
View Forum Message <> Reply to Message

On Tue, 15 Mar 2005, m_schellens@hotmail.com wrote:

> What do you want to hear?
> Obviously the docmentation is wrong here and brackets have a higher
> priority.
> marc
>

Well, I'd like to know whether it is a software bug, a documentation bug,
or simply I am reading something wrong. It is not obvious for me :-)

You voted for documentation bug, thanks.

David's Operator Precedence Tutorial has the same table, and refers to
'[]' and '.' as equal precedence operators, so it is wrong, too :-)

regards,
lajos

Subject: Re: precedence question
Posted by David Fanning on Tue, 15 Mar 2005 14:13:24 GMT
View Forum Message <> Reply to Message

=?ISO-8859-2?Q?F=F6ldy_Lajos?= writes:

> David's Operator Precedence Tutorial has the same table, and refers to
> '[]' and '.' as equal precedence operators, so it is wrong, too :-)

All I know is that as soon as you start parsing "left", "right",
and "equal" people's eyes glaze over. It's right enough for me.
If it's not right *always*, well, I don't hold people to higher
standards than I aspire to. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: precedence question
Posted by Foldy Lajos on Tue, 15 Mar 2005 14:35:29 GMT
View Forum Message <> Reply to Message

On Tue, 15 Mar 2005, David Fanning wrote:

> =?ISO-8859-2?Q?F=F6ldy_Lajos?= writes:
>
>> David's Operator Precedence Tutorial has the same table, and refers to
>> '[]' and '.' as equal precedence operators, so it is wrong, too :-)
>
> All I know is that as soon as you start parsing "left", "right",
> and "equal" people's eyes glaze over. It's right enough for me.
> If it's not right *always*, well, I don't hold people to higher
> standards than I aspire to. :-)
>
> Cheers,
> David
>

Sorry, David, I didn't want to attack you. Your tutorials (and the full
web site) is excellent.

I just came over an expression, where the IDL sw and docs contradict. I
wanted to know, whether this is a real contradiction, or there is a
"hidden rule" somewhere I don't know of.

OK, let's close this topic.

regards,
lajos

Subject: Re: precedence question
Posted by David Fanning on Tue, 15 Mar 2005 14:42:57 GMT
View Forum Message <> Reply to Message

Lajos writes:

> Sorry, David, I didn't want to attack you. Your tutorials (and the full
> web site) is excellent.

Oh, it would take a lot more than that to make me feel
attacked. :-)

> I just came over an expression, where the IDL sw and docs contradict. I
> wanted to know, whether this is a real contradiction, or there is a
> "hidden rule" somewhere I don't know of.

If it is a contradiction between the software and the
documentation, it is the first I have ever discovered
in IDL. :^)

> OK, let's close this topic.

OK. Good idea. :-)

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: precedence question
Posted by JD Smith on Fri, 18 Mar 2005 23:45:54 GMT
View Forum Message <> Reply to Message

On Tue, 15 Mar 2005 14:55:16 +0100, Fï¿½ldy Lajos wrote:

>
> On Tue, 15 Mar 2005, m_schellens@hotmail.com wrote:
>
>> What do you want to hear?
>> Obviously the docmentation is wrong here and brackets have a higher
>> priority.
>> marc
>>
>
> Well, I'd like to know whether it is a software bug, a documentation bug,
> or simply I am reading something wrong. It is not obvious for me :-)
>
> You voted for documentation bug, thanks.
>
> David's Operator Precedence Tutorial has the same table, and refers to
> '[]' and '.' as equal precedence operators, so it is wrong, too :-)

Well, it's actually my tutorial David hosts, and I admit I stole the precedence table straight from the manual without extensive verification. That said, I think you guys have this whole issue wrong.

You're indexing a 3D array using only two of three dimensions. Now you might complain that IDL has irregular behavior in this case, but I don't see it as a precedence issue. Consider instead:

IDL> a=replicate({l:randomu(sd,10)},5)

```
IDL> print,a[2].l[6]
   0.0162049
IDL> print,(a[2]).l[6]
   0.0162049
IDL> print,((a[2]).l)[6]
   0.0162049
```

Looks pretty good. In fact, if you think about it, there is no way for `.' and `[]' to have anything but equal precedence. Let's say the precedence of '[]' really was higher. How would IDL parse a[2].l[6]? Let's see, a[2] is a scalar structure, but now we must first subscript that with [6], since that has higher precedence.... hmmm.

What you've really run into is the apparently variable way IDL treats indexing expressions which specify some, but not all dimensions:

```
IDL> a=findgen(10,10,10)
IDL>  print,a[4,5:6,0]
     54.0000
     64.0000
IDL> print,a[4,5:6]
     54.0000
     64.0000
```

So here IDL just assumes you meant the first plane, as a convenience to you. It could have thrown an error and said "Sorry try again buddy", but it was being friendly instead. The only difference between your case A and case C is that in case A IDL does it's indexing/structure derefencing one by one and builds up the array from the structure, whereas in case C you *first* construct the array of size [2,3,4], and then index it with the incomplete index set [*,0:1], at which point the "assume he meant the first plane" rule for arrays kicks in, and you get the different result. By the way, if you had used

```
IDL> help, a[3].l[*,0:1]
```

instead you would have found consistent behavior as well, since there you're indexing only a 2x3 array.

JD
