
Subject: Re: Assigning structure variables
Posted by [penteado](#) on Thu, 03 Sep 2009 12:27:33 GMT
[View Forum Message](#) <> [Reply to Message](#)

On Sep 3, 9:09 am, Bernhard Reinhardt
<wirdseltengele...@freisingnet.de> wrote:
> pp wrote:
>> On Sep 3, 5:22 am, Bernhard Reinhardt
>> <wirdseltengele...@freisingnet.de> wrote:
>>> Hi,
>
>>> I just ran into a problem when I tried to assign the minimum-index
>>> returned by the min()-function to an array-element I_LON[i].
>
>>> a=min(ABS(GLON - PLON[i]),I_LON[i])
>>> Attempt to store into an expression: <LONG (0)>.
>
>>> IDL> help, I_LON
>>> I_LON LONG = Array[126236]
>
>>> a=min(ABS(GLON - PLON[i]),b)
>>> I_LON[i]=b
>>> works
>
>>> Well, I read Davids tips on precedence and Assigning Structure Values
>>> but they are all about pointers and nested structures. To me it seems
>>> I_LON is a plain array. I guess I haven't yet understood how variables
>>> are exchanged with functions/procedures.
>
>>> Anyone can shed a light on this?
>
>>> Regards
>
>>> Bernhard
>
>> a=min(ABS(GLON - PLON[i]),I_LON[i]) does not work because subscripted
>> arrays are passed to routines by value, instead of by reference. So an
>> argument that is a subscripted array works to pass values to the
>> routine, but the routine cannot pass anything back to it, which is
>> what min is trying to do, and causes the error.
>
>> To say it another way, the error happens because that line is
>> essentially equivalent to
>> a=min(ABS(GLON - PLON[i]),0L), which obviously makes no sense, and
>> that is why IDL complains you are trying to store a value into a
>> constant (<LONG (0)>).
>
>> It is easy to forget it because usually arguments are passed by value

>> in IDL, which is what happens in `a=min(ABS(GLON - PLON[i]),b)`, and is
>> why it works. But subscripted arrays and structure members are passed
>> by value.

>
> Thanks for your explanation. So would you say that

>
> `a=min(ABS(GLON - PLON[i]),b)`

> `I_LON[i]=b`

>
> is an appropriate workaround?

That is the way to do it.
