Subject: Re: Arrays of Structures
Posted by Brian Larsen on Thu, 08 Feb 2007 19:11:54 GMT
View Forum Message <> Reply to Message

Paulv,

I think this is really just a wholly bad idea to do that at all. The reason is what about this structure?

```
IDL> dat=replicate({a:findgen(50)}, 50)
IDL> help, dat
DAT             STRUCT    = -> <Anonymous> Array[50]
IDL> help, dat, /str
** Structure <84ccb14>, 1 tags, length=200, data length=200, refs=1:
   A               FLOAT     Array[50]
```

Here the difference between dat[23].a and dat.a[23] is really obvious.

```
IDL> print, dat[23].a
      0.00000      1.00000      2.00000      3.00000      4.00000
5.00000      6.00000      7.00000      8.00000
      9.00000      10.0000      11.0000      12.0000      13.0000
14.0000      15.0000      16.0000      17.0000
      18.0000      19.0000      20.0000      21.0000      22.0000
23.0000      24.0000      25.0000      26.0000
      27.0000      28.0000      29.0000      30.0000      31.0000
32.0000      33.0000      34.0000      35.0000
      36.0000      37.0000      38.0000      39.0000      40.0000
41.0000      42.0000      43.0000      44.0000
      45.0000      46.0000      47.0000      48.0000      49.0000
IDL> print, dat.a[23]
      23.0000      23.0000      23.0000      23.0000      23.0000
23.0000      23.0000      23.0000      23.0000
      23.0000      23.0000      23.0000      23.0000      23.0000
23.0000      23.0000      23.0000      23.0000
      23.0000      23.0000      23.0000      23.0000      23.0000
23.0000      23.0000      23.0000      23.0000
      23.0000      23.0000      23.0000      23.0000      23.0000
23.0000      23.0000      23.0000      23.0000
      23.0000      23.0000      23.0000      23.0000      23.0000
23.0000      23.0000      23.0000      23.0000
      23.0000      23.0000      23.0000      23.0000      23.0000
```

Be sure to just use the structure correctly and put the [ ] on the thing that is an array whether that is the structure or the thing in the structure (all the dereferencing stuff is true but tends to just confuse the issue in my opinion)

Brian

 -- ---------
Brian A. Larsen
Dept. of Physics
Space Science and Engineering Lab (SSEL)
Montana State University - Bozeman
Bozeman, MT 59717

On Feb 8, 12:02 pm, Paul van Delst <Paul.vanDe...@noaa.gov> wrote:

```
>  Mick Brooks wrote:
>>  On Feb 8, 5:41 pm, "Brian Larsen" <balar...@gmail.com> wrote:
>
>>>  This is what you did and it is true this is an array with 50 elements
>>>  IDL>  HELP, structs.a
>>>  <Expression>    INT       = Array[50]
>
>>>  but if you look at structs by itself not structs.a, it is the array,
>>>  NOT structs.a
>
>>>  IDL>  HELP, structs
>>>  STRUCTS         STRUCT    = -> <Anonymous> Array[50]
>
>>>  So what is inside the structure?
>
>>>  Here a is inside the struct and it is an int
>
>>>  IDL>  HELP, structs, /str
>>>  ** Structure <84bd40c>, 1 tags, length=2, data length=2, refs=1:
>>>     A               INT              1
>
>>  (I think) I'm following you to here...
>
>>>  Meaning that structs.a[23] doesn't make sense because structs.a is an
>>>  int.
>
>>  ... but this is where you lose me. Doesn't the the first line that you
>>  showed say that structs.a is an array of 50 ints?
>>  Maybe I just can't read the output of HELP properly, but struts.a
>>  certainly sometimes behaves as an array. Try PRINTing it, for
>>  instance.
>>  Or let's compare the output of HELP on a real array of 50 ints:
>
>>  IDL> HELP, structs.a
>>  <Expression>    INT       = Array[50]
>>  IDL> HELP, indgen(50)
>>  <Expression>    INT       = Array[50]
```

>
>>> While structs[23].a does make sense because structs is an array.
>
>> Yes, I'm happy with why this is right, but still not clear why
>> structs.a[23] is wrong.
>
> You know, I'm confused now too. Check out the precedence in the IDL help:
>
> Table 12-9: Operator Precedence
>
> 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)
> etc...
>
> So, you see that the the structure field dereference operator, ".", and the array
> subscript operator, "[]" , have the same precedence. Operators with equal precedence are
> evaluated from left to right.
>
> So,
>
>   structs.a[23]
>
> means FIRST dereference the structure field (structs.a), THEN index the array ([23]). The
> way I see it,
>
>   structs.a[23]
> and
>   (structs.a)[23]
>
> should be equivalent.
>
> Maybe the weirdness has something to do with where the result of the dereference "goes" ?
> Hence the "<No name>" in the error message?
>
> Hmm.
>
> paulv
>
> --
> Paul van Delst   Ride lots.
> CIMSS @ NOAA/NCEP/EMC   Eddy Merckx