Subject: Re: Is this a bug?
Posted by menakkis on Thu, 01 Oct 1998 07:00:00 GMT
View Forum Message <> Reply to Message

David Foster <foster@bial1.ucsd.edu> wrote:
<...>
>   ((*p).s.a)[2] = ((*p).s.a)[2] * 4
>   % Temporary variables are still checked out - cleaning up...
>
>   print, ((*p).s.a)[2]
>   48   ; Value was not adjusted

I think something like this came up in the newsgroup a while back, and the problem was that the brackets on the LHS effectively turned it into a RHS, with the REAL LHS becoming a temporary variable.

mmm. I didn't do too well there. How about... You want ((*p).s.a)[2] to receive a value. But IDL sees those outer brackets and decides: "I've got to calculate something here; I'll stick the result in a temporary variable, like I usually do." So you get a temporary variable containing (*p).s.a (or maybe even ((*p).s or *p ???). Finally, IDL assigns your (actual) RHS result to element 2 of the "a" member of this temporary variable, and then chucks away the temporary variable because, as far as it's concerned, the temporary variable has nowhere to go.

Perhaps like me you're a bit uneasy about having to put brackets around dereferenced pointers to get at structure members or array elements therein. It really does go against the grain, doesn't it? Any superfluous brackets and you get this "calculation mode" thing kicking in. I must say that I can't see a way around this, though, given what I gather about how IDL's variables work. I guess an interpretation of what (*p) means is: "make a temporary variable *control* structure to get at the contents of p"; you can then use the "thing" (*p) as if it was a variable name. It takes some getting used-to, but it *is* very efficient for dereferencing just an array element or structure member. (I was going to sound forth on how I dislike this syntax but had second thoughts.)

Peter Mason

----------== Posted via Deja News, The Discussion Network ==----------
http://www.dejanews.com/ Search, Read, Discuss, or Start Your Own