Subject: Re: Functions and arrays
Posted by steinhh on Wed, 04 Dec 1996 08:00:00 GMT
View Forum Message <> Reply to Message

> Please complain directly to RSI about this! We ran into this about a year
> ago, and were unable to get past the first level support folks telling us that
> this was an inevitable consequence of IDL's weak typing.

People at RSI (and not just support people) do read this newsgroup (and act on it as well: XPAD/YPAD/SPACE=0 will once again rule, in IDL v 5.0!), so this place may very well be the best place for a campaign to remove this quite serious (and dangerous) "feature". I do think that we need to show some enthusiasm to make it happen, though (i.e., the more people agreeing about this being a rather nasty thing that they'd like to see removed, the better).

I just cannot see that this is actually an *inevitable* consequence of IDL's typing... it could be an inevitable consequence of a single-pass/single-statement compiler, though. That, however, has nothing to do with the language as such.

Now, assume I want to refer to an element of an array called "test1" somewhere inside a program, e.g.,:

```
value = test1(0)
```

Assuming, of course, that I do *not* write a program that references an undefined variable on purpose, is there *any* way of making this line a non-crashing statement *without* referring to the name "test1" (note: no parentheses) *anywhere* inside the same procedure?

Unless someone can actually show a compelling example of this, I'd say that there is no problem in altering the compiler to behave reasonably, e.g., to automatically detect which names are actually variables, and act accordingly.

Due to the fact that one may write non-crashing programs referencing array elements in statements placed *before* any assignment statement, it's necessary to have either a two-pass compiler or to have some back-tracking for this to work (this must already be done for e.g., the labels in GOTO statements).

If this is too much of a problem, could we at least have a "forward_array" or "forward_variable" statement ? :-)

Or at least get a compile time error about the possible mixup, something like "Error: test1 interpreted as a function in line 5, but as a variable in line 10".