Subject: Re: Q: Efficient Memory handling and deallocation
Posted by peter on Fri, 05 May 1995 07:00:00 GMT
View Forum Message <> Reply to Message

Paul Probert (probert@uwmfe.neep.wisc.edu) wrote:
:
: We've had that problem, and if your program iterates through the "create,
: do something, delete" cycle a few times you run out of memory, because
: the "do something" step inevitably allocates a few more bytes, and these
: come from the hole left by the previous delete. So on the next create
: you don't have a large enough contiguous block of memory. We brought
: this up with the support people at IDL, and they said it was the
: operating system's fault. But we figured out, as you did, that IDL
: doesn't deallocate the memory. One workaround is, at the beginning of
: your program, create and then immediately delete an array 2 or 3 times
: the size of your needs, and this will leave a hole big enough for many
: future reallocations.
:
: Another technique is to give up modular programming and do everything in
: one big main program and never deallocate. But I would really like the
: people at RSI to read these complaints and fix IDL.

Would a C program, using malloc(), perform any differently? That is, if
I had a program that malloc'ed a big block, then free'd it, then
malloc'ed a little block, then started over, would I see the same
behavior as I see from IDL?

If there is no difference, then it really is an OS problem/feature, and
RSI would have to add their own memory management. If not, then maybe
RSI could do something about it. I guess part of the problem is that it
is not usual to use malloc for every little variable in a C program (you
use the stack instead for most scalar variables), whereas IDL used heap
memory for everything (?).

Peter