Subject: Re: deallocating ptrs
Posted by Antonio Santiago on Mon, 13 Jun 2005 06:08:44 GMT
View Forum Message <> Reply to Message

R.G. Stockwell wrote:
> Hi,
> I am reading an HDF, and have a template created with the HDF_browser.
> The template is a structure that has over 200 pointers deeply nested in
> several labrynthical "subnested structure arrays of structure ptr array
> structures".
> :)
>
> So, I tried to deallocate this monster with a heap_free call, and it doesn't
> get
> deallocated. (IDL 6.1)
> In fact I have for the moment been reduced to a heap_gc call, which seems
> inelegant.
>
> So why doesn't heap_free free the heap? It looks like is isn't even trying.
> It goes from 238 to 164 ptrs.
>
> Anyone have a nice generic deallocating routine to crawl down the structure.
> I actually have an old routine that prints out a structure nicely, that
> recursively
> steps through each element. It could be modified to test each element and
> deallocate it if it is a ptr, but there has to be a routine that already
> does that.
>
> Cheers,
> bob
>
>

What about a recursive procedure:

(ALERT: I write next lines directly without testing)

```
PRO free_super_struct_pointer, initial_struct

 num_fields = N_ELEMENTS(TAG_NAMES(initial_struct))
 FOR i=0, num-1 DO BEGIN

    ;;Check if the field is a pointer (to a struct)
    IF SIZE(initial_struct.(i), /TNAME) EQ 'POINTER' THEN BEGIN
  ;;Free nested substructures
        free_super_struct_pointer, *(initial_struct.(i))
```

```
  ;;Free space
        PTR_FREE, initial_struct.(i)
     ENDIF

 ENDFOR
END
```

--

Antonio Santiago Pï¿½rez
(email: santiago<<at>>grahi.upc.edu)
(www: http://www.grahi.upc.edu/santiago)
(www: http://asantiago.blogsite.org)

GRAHI - Grup de Recerca Aplicada en Hidrometeorologia
Universitat Politï¿½cnica de Catalunya

Subject: Re: deallocating ptrs
Posted by R.G. Stockwell on Mon, 13 Jun 2005 20:20:24 GMT
View Forum Message <> Reply to Message

"Antonio Santiago" <d6522117@est.fib.upc.edu> wrote in message
news:d8j7tcabm1@defalla.upc.es...
>
> What about a recursive procedure:
>
> (ALERT: I write next lines directly without testing)
>
>
> PRO free_super_struct_pointer, initial_struct
...
> END
>

Thanks Antonio,
that is what I will end up doing. I just don't understand why
IDL's canned routine heap_free doesn't free the heap. I thought
I must be missing something. After all, if a jarhead like me can
figure out how to deallocate that monstrosity, then surely IDL
should be able to.

Cheers,
bob