Subject: Re: Memory Cleanup-- Messy structure
Posted by rogass on Wed, 16 Jun 2010 21:16:31 GMT
View Forum Message <> Reply to Message

On 16 Jun., 19:50, Ed Hyer <ejh...@gmail.com> wrote:
> OK, here's the sequence:
>
> IDL> STRUCT1=CREATE_STRUCT("PTR1",PTR_NEW());
> IDL> STRUCTALL=REPLICATE(STRUCT1,N)
> IDL> FOR I=0,N-1 DO STRUCTALL.PTR1=PTR_NEW(ALLDATA[*,I])
>
> Then I do what I need to do, and I'm done with it, so:
>
> IDL> UNDEFINE, STRUCTALL
>
> Except this doesn't free the memory, according to HELP,/HEAP.
>
> Since I'm not a real programmer, I still consider pointers to be kinda
> voodoo. Can someone patiently explain how I manually clean up the
> memory in this case?
>
> Thanks,
>
> --Edward H.

Hm, what's going on here. If I would do this on my PC the following
happens:

```
IDL> help,/heap
Heap Variables:
   # Pointer: 0
   # Object : 0
IDL> n=10
IDL> s={ptr1:ptr_new()}
IDL> help,/heap
Heap Variables:
   # Pointer: 0
   # Object : 0
IDL> sall=REPLICATE(S,N)
IDL> help,/heap
Heap Variables:
   # Pointer: 0
   # Object : 0
IDL> FOR I=0,N-1 DO SALL[i].PTR1=PTR_NEW(dist(i+1))
% Compiled module: DIST.
IDL> help,/heap
Heap Variables:
   # Pointer: 10
```

```
   # Object : 0

<PtrHeapVar1>   FLOAT     = Array[1]
<PtrHeapVar2>   FLOAT     = Array[2, 2]
<PtrHeapVar3>   FLOAT     = Array[3, 3]
<PtrHeapVar4>   FLOAT     = Array[4, 4]
<PtrHeapVar5>   FLOAT     = Array[5, 5]
<PtrHeapVar6>   FLOAT     = Array[6, 6]
<PtrHeapVar7>   FLOAT     = Array[7, 7]
<PtrHeapVar8>   FLOAT     = Array[8, 8]
<PtrHeapVar9>   FLOAT     = Array[9, 9]
<PtrHeapVar10>  FLOAT     = Array[10, 10]
IDL> undefine,sall
IDL> help,/heap
Heap Variables:
   # Pointer: 0
   # Object : 0
```

That's why I very like David's routines - they do always their jobs :)

Your problem might be that you don't correctly reference your pointer array. You forgot:
STRUCTALL--->[I]<-----
So, maybe you allocate some memory each time in the loop, but then you always overwrite the pointer(s).

Regards

CR
