
Subject: Re: Releasing memory in IDL

Posted by [Paul Van Delst\[1\]](#) on Wed, 23 Jan 2008 21:15:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

Paul van Delst wrote:

> Jean H wrote:

>> Jaime wrote:

>>> Thanks Jean,

>>>

>>> it helped me... but I was forced to call heap_gc and delete a couple
>>> of arrays (a=0) at the end, but still inside, of the clumpfind
>>> function. After that I still needed to call heap_gc after clumpfind is
>>> called.

>>>

>>> It looks still strange to me that I should all this to get some memory
>>> back... is there a way to avoid the memory leak?

>>>

>>> Best,

>>> Jaime

>>

>> hum...

>> for regular variables (i.e. not pointers), they are released from
>> memory when you exit the pro or function... so setting "a=0" should
>> change the memory used IN the function, but should have no effect when
>> the program returns to a lower / main level. Indeed, this is correct
>> provided that a) the variable is not part of the argument of the
>> function, or b) the variable is not part of a common block.

>>

>> To avoid memory leak... well... pointers need to be well defined and
>> cleaned up!

>>

>> a = ptr_new(indgen(100000000000))

>> a = 0 ==> you just lost track of the indgen(100000000000) array!!! ...

>

> Not completely:

>

> IDL> x=ptr_new(lindgen(100))

> IDL> help, x

> X POINTER = <PtrHeapVar1>

> IDL> help, *x

> <PtrHeapVar1> LONG = Array[100]

> IDL> x=0

> IDL> help, x

> X LONG = 0

> IDL> print, ptr_valid()

> <PtrHeapVar1>

> IDL> y=ptr_valid(1,/cast)

> IDL> help, y

```
> Y          POINTER = <PtrHeapVar1>
> IDL> help, *y
> <PtrHeapVar1> LONG    = Array[100]
>
> Though, off the top of my head, I dunno how you would determine the
> pointer heap variable number programmatically.
```

Duh! I shoulda kept reading the docs before posting....

```
IDL> print, ptr_valid(count=c)
<PtrHeapVar1>
IDL> print, c
      1
```
