Subject: Re: help with structure
Posted by Gray on Wed, 31 Mar 2010 22:12:12 GMT
View Forum Message <> Reply to Message

On Mar 31, 5:49 pm, pp <pp.pente...@gmail.com> wrote:
> On Mar 31, 6:40 pm, David Fanning <n...@dfanning.com> wrote:
>
>
>
>
>
>> Sumit writes:
>>> I want to create a structure with 2 fields. The fields need to be of
>>> variable length. I need to have 'n' such structures,where 'n' is
>>> scalar number. To address the issue of variable length field, I create
>>> pointers for each field(pointing to variable temp for initialization)
>>> as shown below:
>>> mystruct={l1:ptr_new(temp), l2:ptr_new(temp)}
>>> I don't know how to create copies of this structure. Replicate
>>> doesn't work as I guess it creates shallow copy.
>
>> What do you mean it "doesn't work"?
>
>> IDL> mystruct={l1:ptr_new(temp), l2:ptr_new(temp)}
>> IDL> a= replicate(mystruct, 100)
>> IDL> help, a
>> A   STRUCT   = -> <Anonymous> Array[100]
>> IDL> a[50].l2 = Ptr_New(findgen(11))
>
> I suppose he means that by doing that, after the replicate(), all
> elements of a point to the same two heap variables. There is no way
> around it, he needs to loop on the elements to set the pointer on each
> one to point to something new.

Or, he can just create the struct array without allocating heap
variables, then set that field of the array to a ptrarr, and allocate
the heaps then.

```
IDL> mystruct={11:ptr_new(),12:ptr_new()}
IDL> a=replicate(mystruct,100)
IDL> a.11 = ptrarr(100,/allocate_heap)
IDL> a.12 = ptrarr(100,/allocate_heap)
```

Since he's initializing with temporary variables anyway (at least
that's how I read the "temp" in the original post), it doesn't matter
that the heaps aren't initialized doing it this way.