Using IDL 5.4 I get the following:

```
IDL> month_struct = {month_struct, name: ptr_new( ), day: ptr_new( ),
temp_c:ptr_new( )}
IDL> station = {station, number:0L, month:{month_struct}}
IDL> po_basin = replicate (station,10)
IDL> help,po_basin
PO_BASIN      STRUCT    = -> STATION Array[10]
IDL> po_basin.month.name = ptr_new( strarr(2190) )
IDL> help,po_basin.month.name
<Expression>   POINTER   = Array[10]
IDL> help,po_basin[0].month.name
<Expression>   POINTER   = <PtrHeapVar1>
IDL> help,po_basin[1].month.name
<Expression>   POINTER   = <PtrHeapVar1>
```

note that you set all 10 (or howmany ever) pointer to the SAME heap
variable. I assume somehow later therefore you will get an error.

This is a situation were you definitely (objections?) need a for loop:

```
for i=0,9 do po_basin[i].month.name= ptr_new( strarr(2190))
```

Anyway, your snippet works without error.
But you are right,
an object would do the task indeed more elegant/ less error-prone.

hope this helps,
marc

HILBERMAN wrote:
>
> To put it simply, you rock. I have now successfully created a mess: an
> array of a structure
> that contains another embedded structure. Unfortunately, I'm still not
> 'pointed' in the right
> direction. When I try to apply the pointer tip to 'the mess' I get the
> error:
> % Conflicting data structures: <POINTER (<NullPointer>)>,MONTH_STRUCT.
>
> Here's how I have things set up right now.
> month_struct = {month_struct, name: ptr_new( ), day: ptr_new( ), temp_c:
> ptr_new( )}

> station = {station, number:0L, month:{month_struct}}
> po_basin = replicate (station, howmany)
>
> po_basin.month.name = ptr_new(strarr(2190))
> ...
>
> Any ideas? Since station references a structure with pointers, do I have to
> make a pointer to
> station as well--or something similar? I can't say I know a lick about
> objects, but this is
> kinda seeming like a problem to be solved by an object? Oyvey.
>
> Cheers,
> Davida