Subject: Re: [Q] structure definition with variable array size
Posted by Martin Schultz on Thu, 25 Jun 1998 07:00:00 GMT
View Forum Message <> Reply to Message

dEdmundson@Bigfoot.com wrote:
>
> I wish to create a structure for holding a slice from a 3d volume along
> with domain info, e.g.,
>
> pro slice__define
> tmp = {slice, data:fltarr(100,100), x:fltarr(100), y:fltarr(100), z:0.0}
> end
>
> This definition is fine if the data dimensions are known. However, what
> if the array sizes are only known at runtime when the data file is opened
> and brought into IDL. Is it possible to have structures with arbitrary
> length arraysa la F90, C, or Pascal?
>
> Cheers,
> Darran.

Sure is! There are basically two ways (that I know of) to accomplish
this goal:
(1) since IDL does not tag a variable name with a once and forever
defined type information, you can re-create your (anonymous) structure
whenever you need to change something. E.g.:

```
a = -1.
stru = { a:a, b:'none' }   ; float and string
a = fltarr(100)
stru = { a:a, b:n_elements(a) }  ; float array and int
; note that struct.a = a would have failed !
```

(2) you can use pointers in your structure instead of the actual array:

```
; initialize dummy structure
stru = { a:ptr_new(), b:-1 }
; create pointer to data array
a = ptr_new(fltarr(100))
stru.a = a
stru.b = n_elements(*a)
help,stru,/stru   ; gives you some information
help,*stru.a      ; derefences the pointer

; don't forget to free pointer at the end:
if (ptr_valid(stru.a)) then begin
    ptr_free,stru.a
    stru.b = -1  ; in case you access stru later
endif
```

; ... or do a garbage collection when you exit your program
remains = ptr_valid(count=count)
if (count gt 0) then ptr_free,remains

Hope this helps,
Martin.

--
---------- -------
Dr. Martin Schultz
Department for Earth&Planetary Sciences, Harvard University
109 Pierce Hall, 29 Oxford St., Cambridge, MA-02138, USA

phone: (617)-496-8318
fax : (617)-495-4551

e-mail: mgs@io.harvard.edu
Internet-homepage: http://www-as.harvard.edu/people/staff/mgs/
---------- -------

Subject: Re: [Q] structure definition with variable array size
Posted by Matthew J. Sheats on Thu, 25 Jun 1998 07:00:00 GMT
View Forum Message <> Reply to Message

> I wish to create a structure for holding a slice from a 3d volume along
> with domain info, e.g.,
>
> pro slice__define
> tmp = {slice, data:fltarr(100,100), x:fltarr(100), y:fltarr(100), z:0.0}
end
>
> This definition is fine if the data dimensions are known. However, what
> if the array sizes are only known at runtime when the data file is opened
> and brought into IDL. Is it possible to have structures with arbitrary
> length arraysa la F90, C, or Pascal?
>
> Cheers,
> Darran.

Unfortunately, the only way I have found to defeat this limitation is
to use pointers. For example:

tmp = { slice, data:PTR_NEW(0), x:PTR_NEW(0), ... } etc for all dynamic

arrays.

Then in run time...

data = PTR_NEW(FLTARR(100,100))

Now that just actually allocates the pointer, you haven't actually allocated the array yet, you need one more call like this:

(*data) = FLTARR(100,100)

Now you have a dynamically set array in a structure. Bit of a pain, but it works. And it works great with IDL's OOP. Just remember you have to use the pointer dereference whenever you access the array:

(*data)[10,15] = 5.0

or whatever.

Hope this help,

Matthew Sheats
Los Alamos National Laboratory

Subject: Re: [Q] structure definition with variable array size
Posted by davidf on Sun, 28 Jun 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Matthew J. Sheats (sheats@lanl.gov) writes:

>> I wish to create a structure for holding a slice from a 3d volume along
>> with domain info, e.g.,
>>
>> pro slice__define
>> tmp = {slice, data:fltarr(100,100), x:fltarr(100), y:fltarr(100), z:0.0}
>> end
>>
>> This definition is fine if the data dimensions are known. However, what
>> if the array sizes are only known at runtime when the data file is opened
>> and brought into IDL. Is it possible to have structures with arbitrary
>> length arraysa la F90, C, or Pascal?
>>
>
> Unfortunately, the only way I have found to defeat this limitation is
> to use pointers.

"Unfortunately"!? Fortunately, there *are* pointers to make these

kinds of things possible. Think of a world without arrays of variable length vectors, etc. Why, we would all have to learn to program in BASIC. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting
E-Mail: davidf@dfanning.com
Phone: 970-221-0438
Coyote's Guide to IDL Programming: http://www.dfanning.com/
