Subject: Re: Starting a for loop within an if loop
Posted by Allan Whiteford on Thu, 20 Dec 2007 09:31:40 GMT
View Forum Message <> Reply to Message

Mat Smith wrote:
> On Dec 20, 1:14 am, David Fanning <n...@dfanning.com> wrote:
>
>> Mat Smith writes:
>>
>>> I'm trying to write a program with some keywords, so that depending
>>> upon my mood I can alter what it does.
>>
>>> What I want to do is most of the time just run the program, but I want
>>> to include an option to do a for loop.
>>
>>> Basically, I need something like:
>>
>>> IF keyword_set(mc) THEN BEGIN
>>> FOR i=0,n-1 DO BEGIN
>>> t=d[i]
>>> ENDIF
>>
>>> and later in the program
>>
>>> IF keyword_set(mc) THEN BEGIN
>>> ENDFOR
>>> ENDIF
>>
>>> Thus the rest of the program runs normally with and without the for
>>> loop (it just uses t).
>>> However, IDL doesn't understand this - it wants an ENDFOR before the
>>> ENDIF.
>>
>>> Any thoughts on how I can get around this? It's entirely possible that
>>> I'm missing something obvious.
>>
>> It's possible. Have you taught yourself to program?
>> Sometimes there can be gaps. :-)
>>
>> Anyway, I think I would do something like this:
>>
>> IF keyword_set(mc) THEN endloop = n ELSE endloop = 1
>> FOR j=0,endloop-1 DO BEGIN
>> ...
>> ENDFOR
>>
>> This way, if your keyword is set, you will do the loop n times,
>> otherwise you will do the loop just once.

>>
>> Cheers,
>>
>> David
>> --
>> David Fanning, Ph.D.
>> Fanning Software Consulting, Inc.
>> Coyote's Guide to IDL Programming:http://www.dfanning.com/
>> Sepore ma de ni thui. ("Perhaps thou speakest truth.")
>
>
> Hi David,
>
> I have indeed taught myself to program, and there are undoubtedly huge
> gaps!! I've never been one for learn the basics - I'm more of a dive
> in and solve problems as they come!
>
> If I read your reply correctly, I don't think that it'll do what I
> want it to, but I can adapt it so that it will!
>
> The problem was that I wanted to have different structures depending
> on if the keyword is set.
>

You might want to look at making your "... working on t" a subroutine in
its own right, this would give you code like:

```
pro doit,t
 ... work on t
end

if keyword_set(mc) then begin
 for i=0,k-1 do begin
  doit,d[i]
 end
endif else begin
 doit,s
endelse
```

it will give you slightly more flexibility to put your core code inside
different structures etc. If mc stands for Monte Carlo then you almost
certainly want to separate your model from the controlling code.

I'd guess you'd also want to pass an answer back out of "doit" in which
case you might want to make it a function or add more parameters.

> That is if the keyword isn't set then it will have a particular
> values, but if it is set, then it will run the loop AND the values

> will always be different to when the keyword isn't set (i.e. loop of 1
> isn't the not keyword).
>
> So before I had
>
> IF NOT keyword_set(mc) THEN BEGIN
> t=s
> ENDIF ELSE BEGIN
> FOR i=0,k-1 DO BEGIN
> t=d[i]
> ENDELSE
> ... working on t
> ENDFOR - which obviously didn't work
>

It's not a completely unreasonable thing to want - it's just that IDL
won't let you do it.

> But I can adapt your suggestion to make it work - i.e. in the for loop
> call another if loop! e.g.
>
> IF keyword_set(mc) THEN endloop = n ELSE endloop = 1
> FOR j=0,endloop-1 DO BEGIN
> IF NOT keyword_set(mc) THEN BEGIN
> t=s[j]
> ENDIF ELSE BEGIN
> t=d[j]
> ENDELSE
> ...woking on t
> ENDFOR
>
> Does this make sense? It's very long-winded (there must be a quicker
> way!). I'm trying to be clever and I'm probably making my programs too
> generic. I'm also probably missing something simple (thanks to the
> learning it as I go!)

In this context, you'd be better with something like:

```
if keyword_set(mc) then begin
 endloop=n
 t=d ; d can be an array
endif else begin
 endloop=1
 t=s ; s can be a scalar
endelse

for j=0,endloop-1 do begin
 ; now work on t[i]
```

end

this will save doing an "if" inside of your "for". IDL will not optimise this away for you like some other compilers might claim to.

Thanks,

Allan
