Subject: Re: is this a bug in IDL?
Posted by Wonko[3] on Tue, 16 Dec 2003 10:58:00 GMT
View Forum Message <> Reply to Message

aramisgm@hotmail.com (Aramis Martinez) wrote:

> k=0
> for i=0,8759 do begin
> x=vbs(2,i)
> if x gt 0.0 then k=k+1 & x=0 & vbs(2,i)=x
> endfor
> print, k

Which is the same as:

```
k=0
for i=0,8759 do begin
        x=vbs(2,i)
        if x gt 0.0 then k=k+1
        x=0
        vbs(2,i)=x
endfor
print, k
```

> 1) How would you vectorize this? I tried a few things that I could
> remember from my work this summer, but four months of quantum
> mechanics wipes a lot from memory :)

This should do:

```
index = where( vbs[ 2, lindgen( 8760 ) ] gt 0, k )
if k gt 0 then vbs[ 2, index ] = 0
```

[...]

> Is this a bug? as far as I can tell, the statements are logically
> identical since an &'s behavior should never vary, so what in the guts
> of IDL would make these statements be different? Is there some rule
> kicking in here that's not so obvious but makes this behavior to be
> expected? Would parenthesis around the THEN block make these identical
> in IDL? The environment is either IDL 6 on Red Hat 9 or IDL 5.1 on
> Red Hat 8.1.

Using & ist the same as wrting the next statement in another line. You
need to use begin and endif statements.

Alex
--
Alex Schuster Wonko@wonkology.org PGP Key available
alex@pet.mpin-koeln.mpg.de

Subject: Re: is this a bug in IDL?
Posted by Ben Panter on Tue, 16 Dec 2003 11:09:13 GMT
View Forum Message <> Reply to Message

Aramis Martinez wrote:
> k=0
> for i=0,8759 do begin
> x=vbs(2,i)
> if x gt 0.0 then k=k+1 & x=0 & vbs(2,i)=x
> endfor
> print, k
>
> k=0
> for i=0,8759 do begin
> x=vbs(2,i)
> ;; version A
> if x gt 0.0 then x=0 & k=k+1 & vbs(2,i)=x
> ;; version B
> if x gt 0.0 then k=k+1 & x=0 & vbs(2,i)=x
> endfor
> print, k
>

I'm no expert, but I think that possibly your code is using & in the wrong way - as far as I understand it (correct me if I'm wrong guys!!) & means "complete the previous command then do the thing after me" - so your code actually reads:

```
k=0
for i=0,8759 do begin
   x=vbs(2,i)
   if x gt 0.0 then x=0
   k=k+1
   vbs(2,i)=x
   endfor
```

or

```
k=0
for i=0,8759 do begin
   x=vbs(2,i)
   if x gt 0.0 then k=k+1
```

```
   x=0
   vbs(2,i)=x
   endfor
```

To get the behaviour that you want I think you need another begin...end set:

```
k=0
for i=0,8759 do begin
   x=vbs(2,i)
   if x gt 0.0 then begin
 k=k+1
  x=0
    vbs(2,i)=x
 end
   endfor
```

--
Ben Panter, Edinburgh
My name (no spaces)@bigfoot which is a com.

Subject: Re: is this a bug in IDL?
Posted by aramisgm on Tue, 16 Dec 2003 21:17:28 GMT
View Forum Message <> Reply to Message

I've used index, but I'd forgotten about it. Thanks for the reminder.

Ah, the & means for each iteration do an if-then block, then do the two following statements (as opposed to tying together the three statements as the then part of the if-then block) -- the scope for the & is slightly different than in, say, Java. However, now there is a new problem. The entire array should get zeroed out if the snippet below is what is happening. . . (five minutes and a phone call later) indeed it is! To me this seems to be a non-standard usage of &, but that's what I get for being trained in c and its descendants :+) The three statements go into a do-begin-end block, problem solved.

Thank you gentlemen. Enlightenment received.
Aramis Martinez

Wonko@wonkology.org (Alex Schuster) wrote in message news:<8zysEJaud8B@wonkology.org>...
> aramisgm@hotmail.com (Aramis Martinez) wrote:
>
>> k=0
>> for i=0,8759 do begin

>> x=vbs(2,i)
>> if x gt 0.0 then k=k+1 & x=0 & vbs(2,i)=x
>> endfor
>> print, k
>
> Which is the same as:
>
> k=0
> for i=0,8759 do begin
> x=vbs(2,i)
> if x gt 0.0 then k=k+1
> x=0
> vbs(2,i)=x
> endfor
> print, k
>
>
>> 1) How would you vectorize this? I tried a few things that I could
>> remember from my work this summer, but four months of quantum
>> mechanics wipes a lot from memory :)
>
> This should do:
>
> index = where(vbs[2, lindgen(8760)] gt 0, k)
> if k gt 0 then vbs[2, index] = 0
>
>
> [...]
>
>> Is this a bug? as far as I can tell, the statements are logically
>> identical since an &'s behavior should never vary, so what in the guts
>> of IDL would make these statements be different? Is there some rule
>> kicking in here that's not so obvious but makes this behavior to be
>> expected? Would parenthesis around the THEN block make these identical
>> in IDL? The environment is either IDL 6 on Red Hat 9 or IDL 5.1 on
>> Red Hat 8.1.
>
> Using & ist the same as wrting the next statement in another line. You
> need to use begin and endif statements.
>
> Alex