Subject: Re: FOR loop ends one above where it's supposed to
Posted by mankoff on Wed, 02 Jun 2010 00:53:21 GMT
View Forum Message <> Reply to Message

On Jun 1, 5:26 pm, Bulgakovv <alex.ph.sjog...@gmail.com> wrote:
> Maybe there are previous posts about this, but I tried to google
> around without any success so I'll throw the question out.
>
> When I make a FOR loop from for example 0 to 5, the counter ends on
> the value one step above what I stated. Here's an example:
>
>> for i =0,5 do print,i
>
> 0
> 1
> 2
> 3
> 4
> 5>print,i
>
> 6
>
> Is this how it's supposed to be in IDL? To me it seems a little bit
> weird, why would the FOR loop count one extra step in the end? Took me
> a couple of hours to realize what was going on in my code as I took
> for granted this was not the case, so now I'm interested in
> understanding why IDL is behaving like this. Thanks and sorry if
> reposting!

The way I think of for loops is this:
>> for i =0,5 do print,i
1) Set i = 0
2) Check if i gt 5
3) If not, do the loop
4) increment i
5) Go to step (2)

So, step 4 happens incrementing from 5 to 6 at the end of the loop,
and then it continues past the end of the loop, but no loop events
occur with i equal to 6. Every other language I have used behaves
analogous to this, I think.

-k.

Subject: Re: FOR loop ends one above where it's supposed to
Posted by cameron bowles on Wed, 02 Jun 2010 05:34:23 GMT

View Forum Message <> Reply to Message

IDL is a little different to other languages in this regard and it
handles for loops in this manner;

1. generates the index variable (first value, in your case i=0)
2. generates limit value (last value, in your case 5, stored as a
temporary variable)
3. generates step value (in your case 1, stored as a temporary
variable)
4. checks if limit is greater than index (for positive step values, or
less than for negative increments), if so then FOR loop finishes.
5. the block of statements following the DO is executed.
6. the step value is added to the index
7. repeat steps 4->6 until step 4 fails.

the difference here is IDL only uses an index variable, a lot of other
languages uses a secondary temporary value to do the FOR conditional
check and then updates the index only if the FOR condition is
satisfied.

hope that helps,
Cam

Subject: Re: FOR loop ends one above where it's supposed to
Posted by cameron bowles on Wed, 02 Jun 2010 05:36:51 GMT
View Forum Message <> Reply to Message

On Jun 2, 2:34 pm, cameron bowles <cameronbowle...@gmail.com> wrote:
> IDL is a little different to other languages in this regard and it
> handles for loops in this manner;
>
> 1. generates the index variable (first value, in your case i=0)
> 2. generates limit value (last value, in your case 5, stored as a
> temporary variable)
> 3. generates step value (in your case 1, stored as a temporary
> variable)
> 4. checks if limit is greater than index (for positive step values, or
> less than for negative increments), if so then FOR loop finishes.
> 5. the block of statements following the DO is executed.
> 6. the step value is added to the index
> 7. repeat steps 4->6 until step 4 fails.
>
> the difference here is IDL only uses an index variable, a lot of other
> languages uses a secondary temporary value to do the FOR conditional
> check and then updates the index only if the FOR condition is
> satisfied.

>
> hope that helps,
> Cam

I meant step 4 to read;
4. checks if index is greater than limit (for positive step values, or
less than for negative increments), if so then FOR loop finishes.

Subject: Re: FOR loop ends one above where it's supposed to
Posted by pgrigis on Wed, 02 Jun 2010 15:57:21 GMT
View Forum Message <> Reply to Message

On Jun 1, 8:26 pm, Bulgakovv <alex.ph.sjog...@gmail.com> wrote:
> Maybe there are previous posts about this, but I tried to google
> around without any success so I'll throw the question out.
>
> When I make a FOR loop from for example 0 to 5, the counter ends on
> the value one step above what I stated. Here's an example:
>
>> for i =0,5 do print,i
>
> 0
> 1
> 2
> 3
> 4
> 5>print,i
>
> 6
>
> Is this how it's supposed to be in IDL? To me it seems a little bit
> weird, why would the FOR loop count one extra step in the end? Took me
> a couple of hours to realize what was going on in my code as I took
> for granted this was not the case, so now I'm interested in
> understanding why IDL is behaving like this. Thanks and sorry if
> reposting!

While people here gave a logical explanation as to what is happening,
I would say that the *safest* practice in your programming is to not
use the for loop index variable ("i" in this case) for anything else
in your program. That will make the code much easier to write, read
and understand.

Ciao,
Paolo

Subject: Re: FOR loop ends one above where it's supposed to
Posted by penteado on Wed, 02 Jun 2010 16:07:44 GMT
View Forum Message <> Reply to Message

On Jun 2, 12:57 pm, Paolo <pgri...@gmail.com> wrote:
> While people here gave a logical explanation as to what is happening,
> I would say that the *safest* practice in your programming is to not
> use the for loop index variable ("i" in this case) for anything else
> in your program. That will make the code much easier to write, read
> and understand.

Which is why I miss block scope in IDL, so that the variable would not
even exist outside the loop.

Subject: Re: FOR loop ends one above where it's supposed to
Posted by Bulgakovv on Wed, 02 Jun 2010 16:49:04 GMT
View Forum Message <> Reply to Message

On Jun 2, 12:07 pm, pp <pp.pente...@gmail.com> wrote:
> On Jun 2, 12:57 pm, Paolo <pgri...@gmail.com> wrote:
>
>> While people here gave a logical explanation as to what is happening,
>> I would say that the *safest* practice in your programming is to not
>> use the for loop index variable ("i" in this case) for anything else
>> in your program. That will make the code much easier to write, read
>> and understand.
>
> Which is why I miss block scope in IDL, so that the variable would not
> even exist outside the loop.

Thanks for your comments! In Matlab for example, the loop index
variable stays on the last number, in this case 5, I guess that's how
I confused it in first place as I'm used to Matlab. I used the loop
index variable for testing in my program, not to be in there, that's
how I came across the whole situation. But, I guess I will have to
follow your advice Paolo..