Subject: Re: array subscripts
Posted by Benjamin Hornberger on Mon, 14 Nov 2005 15:26:48 GMT
View Forum Message <> Reply to Message

sebinjapan wrote:
> Hi,
>
> Hi,
>
> I found a strange feature when using out of bounds subscripts.
> I am using IDL 6.2 for Mac OSX
> there is a short example:
>
> IDL> a=indgen(4)
> IDL> print, a[-1]
> % Attempt to subscript A with <INT (-1)> is out of range.
> % Execution halted at: $MAIN$
>
> that's ok, but that happens if we use the array [-1] instead of the scalar
> -1 as the subscript address
>
> IDL> print, a[[-1]]
> 0
>
> well... looks like print, a[0 > [-1]]
>
> Same idea for "too large" subscripts
> IDL> print, a[4]
> % Attempt to subscript A with <INT (4)> is out of range.
> % Execution halted at: $MAIN$
> IDL> print, a[[4]]
> 3
>
> IDL> print, a[indgen(8)-2]
> 0 0 0 1 2 3 3 3
>
>
>

From the IDL help:

"Clipping
If an element of the subscript array is less than or equal to zero, the first element of the subscripted array is selected. If an element of the subscript array is greater than or equal to the last subscript in the subscripted array, the last element is selected.

Note
Elements of the subscript array that are negative or larger than the highest subscript are clipped to the target array boundaries. Note that a common error is to use a negative scalar subscript (e.g., A[-1]). Using this type of subscript causes an error. Negative array subscripts (e.g., A[[-1]]) do not cause errors.

This clipping of out of bounds elements can be disabled within a routine by using the STRICTARRSUBS option to the COMPILE_OPT statement. (See the documentation for COMPILE_OPT for details.) If STRICTARRSUBS is in force, then array subscripts that refer to out of bounds elements will instead cause IDL to issue an error and stop execution, just as an out-of-range scalar subscript does. "

Just had to read all that myself last night ...

Benjamin
