Subject: Re: Array subscripting oddity
Posted by harris on Tue, 17 Aug 1993 15:47:54 GMT

View Forum Message <> Reply to Message

glenn@atm.ch.cam.ac.uk (Glenn Carver) writes:

> I thought I understand array subscripting in PV-wave but this bit of
> code has got me foxed. I was trying to figure out what was going wrong
> with a procedure and finally tracked it down to a problem that I can
> demonstrate using the following example:

```
> WAVE> a=indgen(3)
> WAVE> print,a
>        0       1       2
> WAVE> m=[45,99]
> WAVE> print,m
>       45      99
> WAVE> a(m)=255
> WAVE> print,a
>        0       1     255
```

> Now, I would have thought that because both elements of 'm' are more
> than the size of 'a', no element of 'a' would have been assigned?
> The values '45' and '99' are completely arbitrary. You can set them to
> anything you like.

> If someone could explain what's going on here (maybe it's a bug??) I'd
> be most grateful. The configuration is PV-wave 4.01 running on SunOS
> 4.1.3.

```
>       Thanks.

> ---
>  ----------------------------------------------------------- ------------
> Dr Glenn Carver                         Email: glenn@atm.ch.cam.ac.uk
> Centre for Atmospheric Science          Phone: (44-223) 336521
> Chemistry Department                    Fax  : (44-223) 336362
> Cambridge University
> UK
>  ----------------------------------------------------------- ------------
```

Yes, you found a bug. VNI knows about it and it should be fixed shortly.

Mitch

```
--
Mitch Harris                        |  New Technology Inc.
(205)-544-2846                      |  700 Boulevard South, #401
drvax3!harris@freedom.msfc.nasa.gov |  Huntsville, AL  35802
```

Subject: Re: Array subscripting oddity
Posted by offenbrg on Tue, 17 Aug 1993 16:40:45 GMT
View Forum Message <> Reply to Message

glenn@atm.ch.cam.ac.uk (Glenn Carver) writes:

```
> WAVE> a=indgen(3)
> WAVE> print,a
>        0       1       2
> WAVE> m=[45,99]
> WAVE> print,m
>       45      99
> WAVE> a(m)=255
> WAVE> print,a
>        0       1     255
```

> Now, I would have thought that because both elements of 'm' are more
> than the size of 'a', no element of 'a' would have been assigned?
> The values '45' and '99' are completely arbitrary. You can set them to
> anything you like.

I'm speaking from IDL experience, not PV-WAVE, but here goes:

IDL isn't like C. In C, if A is allocated as a 3-element vector, you can still read from or write to A[45] (albeit with unpredictable results :).

In IDL an array isn't exactly a pointer like it is in C. If you try to access an element outside of the range of the array, you get an error message. For example, if you said "a(45) = 255", you'd get an "array subscript out of range" (or some such) message.

When the subscript (m in the example) is a vector, IDL treats it differently. Essentially, it treats out-of-bounds elements *when the subscript is a vector* as if they were the nearest element (either element 0 or the last element).

Example:

```
IDL> a = indgen(3) & print,a
 0 1 2
IDL> print, a(-1) & print, a(5)
% Attempt to subscript A with <INT      (      -1)> is out of range.
% Execution halted at $MAIN$  .
% Attempt to subscript A with <INT      (       5)> is out of range.
% Execution halted at $MAIN$  .

IDL> print, a([-1,5]) ;The subscript is now a vector
 0 2
```

I'm not quite sure why it was done this way--I think it is because one can

treat A as a function, and this way A doesn't return null values, and the
result has the same number of elements as the input.
e.g.

```
IDL> F = A(m)  ;F has the same dimensions as m, even if individual elements
        ;of m are out of range for A.
```

Joel

--
Joel D. Offenberg |
Hughes STX / UIT Science Team | I get paid to stare into space.
offenbrg@fondue.gsfc.nasa.gov |
Don't hold anyone responsible for what I say. I'm on my coffee break.

Subject: Re: Array subscripting oddity
Posted by kevin on Wed, 18 Aug 1993 00:14:19 GMT

View Forum Message <> Reply to Message

glenn@atm.ch.cam.ac.uk (Glenn Carver) writes:

> I thought I understand array subscripting in PV-wave but this bit of
> code has got me foxed. I was trying to figure out what was going wrong
> with a procedure and finally tracked it down to a problem that I can
> demonstrate using the following example:

```
> WAVE> a=indgen(3)
> WAVE> print,a
>       0       1       2
> WAVE> m=[45,99]
> WAVE> print,m
>      45      99
> WAVE> a(m)=255
> WAVE> print,a
>       0       1     255
```

> Now, I would have thought that because both elements of 'm' are more
> than the size of 'a', no element of 'a' would have been assigned?
> The values '45' and '99' are completely arbitrary. You can set them to
> anything you like.

Looks like a bug to me. the values of m are not quite arbitrary. It works
right if these values are in the proper range for the size of a

Kevin Anderson

Subject: Re: Array subscripting oddity
Posted by jdlb on Thu, 19 Aug 1993 01:25:28 GMT

View Forum Message <> Reply to Message

glenn@atm.ch.cam.ac.uk (Glenn Carver) writes:
[Sample array indexing problem, showing strange behavior:]

```
glenn> WAVE> a=indgen(3)
glenn> WAVE> m=[45,99]
glenn> WAVE> a(m)=255
glenn> WAVE> print,a
glenn>       0       1     255
```

glenn> Now, I would have thought that because both elements of 'm' are more
glenn> than the size of 'a', no element of 'a' would have been assigned?

kevin@dipl.rdd.lmsc.lockheed.com (Kevin Anderson) responds:
kevin> Looks like a bug to me.

I don't know about PV-WAVE, but in IDL it's not really a bug--it's a strange but documented feature. From the IDL v. 3.0 manual:

Page 5-5, in "Array Subscripts" section:

> If an element of the subscript array [M in the example above] is greater than or equal to the last subscript in the subscripted variable [A, above], the last element [of A] is selected.

Similarly, a negative subscript refers to the first (0th) element of the array. The same information is expressed in different words on p. 6-4, in subsection "Using Array Subscripts with the Second Form of the Assignment Statement."

It would be more logical, and easier to avoid errors, if this "feature" were not present. After all, a scalar subscript causes an error if out of bounds. And a statement like

```
  b(where(b EQ c)) = d
```

makes b(0)=d if there are no elements where b=c, because the WHERE statement returns -1 if no elements are found.

glenn> The values '45' and '99' are completely arbitrary. You can set them to
glenn> anything you like.

kevin> the values of m are not quite arbitrary. It works right if
kevin> these values are in the proper range for the size of a.

Like Glenn, I was able to reproduce this behavior with arbitrary values of M.

--Jeff

% J-F de La Beaujardiere % jdlb@ifa.hawaii.edu %

```
% Institute for Astronomy %     808-956-9843    %
%  University of Hawai`i  %     fax 956-9402    %
```

Subject: Re: Array subscripting oddity
Posted by sterner on Fri, 20 Aug 1993 14:17:52 GMT
View Forum Message <> Reply to Message

glenn@atm.ch.cam.ac.uk (Glenn Carver) writes:

> I thought I understand array subscripting in PV-wave but this bit of
> code has got me foxed. I was trying to figure out what was going wrong
> with a procedure and finally tracked it down to a problem that I can
> demonstrate using the following example:

```
> WAVE> a=indgen(3)
> WAVE> print,a
>        0       1       2
> WAVE> m=[45,99]
> WAVE> print,m
>       45      99
> WAVE> a(m)=255
> WAVE> print,a
>        0       1     255
```

> Now, I would have thought that because both elements of 'm' are more
> than the size of 'a', no element of 'a' would have been assigned?
> The values '45' and '99' are completely arbitrary. You can set them to
> anything you like.

> If someone could explain what's going on here (maybe it's a bug??) I'd
> be most grateful. The configuration is PV-wave 4.01 running on SunOS
> 4.1.3.

IDL does the same thing. I think of it as a feature, not a bug.
You can use it to your advantage as follows. Let x be a 10 element
array. t = x(i) will fail for scalar values of i that are less than
0 or greater than 9. It is not uncommon that you want to
force i to be in the correct subscript range, in this case you
could say t = x(i>0<9) but in general it is extra work to find
the upper subscript limit. A much simpler fix is t = x([i]).

Ray Sterner   sterner@tesla.jhuapl.edu
Johns Hopkins University   North latitude 39.16 degrees.
Applied Physics Laboratory   West longitude 76.90 degrees.
Laurel, MD 20723-6099