Subject: Array with float indices.
Posted by Nikola on Fri, 21 Jun 2013 12:01:20 GMT
View Forum Message <> Reply to Message

I just discovered a bug in the code that I'm writing. It was caused by a nasty feature of IDL that I wasn't aware before. This behaviour - maybe naively - I would call erratic.

Briefly, I use an array of indeces (xind) to extract a subarray (y) from an array (x). Something like:

```
x = FINDGEN(5)/5
xind =[2, 3, 4]
y = x[xind]
```

The indeces xind are computed by a subroutine that gives a FLOAT output. It may happen that the indeces are out of the range of the original array (x), e.g. xind = [3, 4, 5]. I was naively assuming that my code would crash on that. Nevertheless, it turned out that the code can live with that as long as the indeces are submitted as a float array with arbitrary values?!? Using the same example as above:

```
print, x[5]
% Attempt to subscript X with <INT       (       5)> is out of range.
% Execution halted at: $MAIN$

print, x[5.]
% Attempt to subscript X with <FLOAT     (      5.00000)> is out of range.
% Execution halted at: $MAIN$

print, x[[5.]]
    0.800000

print, x[[15.]]
    0.800000
```

Is this actually a feature of IDL? Is it documented somewhere? I found it to be quite dangerous as a potential source of bugs. I expected that

(1) x[any scalar] = x[[any scalar]]
(2) x[float scalar] = x[FLOOR(float scalar) or ROUND(float scalar)]

but not
x[[float]] = x[float < N_ELEMENTS(x)-1]

Note: It's common for 7.1 and 8.2.3.