Subject: Re: Feature request: printing very long arrays
Posted by Helder Marchetto on Wed, 10 Jun 2015 07:56:13 GMT
View Forum Message <> Reply to Message

On Tuesday, June 9, 2015 at 5:57:20 PM UTC+2, fawltyl...@gmail.com wrote:
> On Tuesday, June 9, 2015 at 3:43:33 PM UTC+2, john.c...@gmail.com wrote:
>> On Monday, June 8, 2015 at 8:48:47 AM UTC-4, Helder wrote:
>>> Hi,
>>> I don't know if this happens only to me, but sometimes while debugging I like to look at what's inside a variable. Most of the times I use the command:
>>>
>>> help, variable
>>>
>>> and sometimes
>>>
>>> print, variable
>>>
>>> However, sometimes I'm too eager to look at what's hidden under the name and I go directly for the print option. And if I'm so stupid to do that on array of say 4096 x 4096 elements... well it takes a while and the only way to stop this useless overflow of data is to kill the IDL process.
>>>
>>> Is there a chance we a print command that looks like this:
>>>
>>> IDL> print, veryBigVariable
>>> [ 0 1 ... 999998 999999]
>>>
>>> and
>>> IDL> print, veryBigVariable, /fullPrint
>>> 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14
15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
31 32
>>> 33 34 35 36 37 38 39 40 41 42 43 44 45 46
47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62
63 64 65
>>> ....
>>>
>>> well you got the point.
>>>
>>> Any chance of this showing up in the future?
>>>
>>> Cheers,
>>> Helder
>>
>>
>> +1 to this request, as I have made the same mistake too many times to count. I would guess there are backwards compatibility issues here though.
>>
>> John

>
> You can write your own print procedure, something like:
>
> pro myprint, x, fullprint=full
> help, x
> n=n_elements(x)
> if n le 10 or keyword_set(full) then print, x $
> else print, x[0:4], '...', x[n-5:n-1]
> end
>
> regards,
> Lajos

Thanks Lajos,
I didn't think of that easy solution... Just made my "p" (=print) like this:

```
pro p, inVar, fullprint=fullprint
n=size(inVar)
if n[-1] eq 0 then print, 'variable undefined' $
else begin
  if n[-1] le 10 || keyword_set(fullprint) then print, inVar $
  else begin
    if n[0] eq 1 then f = '(i0)' $
                else f = '('+strtrim(n[0]-1,2)+'(i0,","),'+'(i0))'
    print, 'variable has '+strtrim(n[0],2)+' dimensions with ('+string(n[1:-3], format=f)+') elements
and a total of '+strtrim(n[-1],2)+' elements'
    print, inVar[0:1], '...', inVar[n[-1]-2:n[-1]-1]
  endelse
endelse
end
```

Not very elegant, but does the job.

Cheers,
Helder