Subject: Re: Succinct way of testing array membership Posted by DavidF[1] on Thu, 30 Aug 2012 16:57:02 GMT

View Forum Message <> Reply to Message

1	⊇.	$\overline{}$	٦	h	е	r١	۸,	ri	+	$\overline{}$	c	•
ι	7	u	( 1	( )	$\leftarrow$	I١	w	п	ш	H	5	

> Is there a more succinct way of testing array membership than using where and n\_elements on the indexes? Something like array\_contains(a, 'pancakes')?

It might take you 30 seconds to write such a thing!

Cheers.

David

Subject: Re: Succinct way of testing array membership Posted by Russell Ryan on Thu, 30 Aug 2012 16:59:43 GMT View Forum Message <> Reply to Message

On Thursday, August 30, 2012 12:51:48 PM UTC-4, godber wrote:

> Is there a more succinct way of testing array membership than using where and n\_elements on the indexes? Something like array contains(a, 'pancakes')?

>

>

> Austin

wouldn't that just be liek this:

suppose:

t=['pancakes','eggs','milk','sausage'] ;array to test

then:

g=where(strmatch(t,'pancakes',/fold),n); the line testing it if n ne 0 then print,'lts in the array!'

-russell

Subject: Re: Succinct way of testing array membership Posted by Michael Galloy on Thu, 30 Aug 2012 17:09:33 GMT View Forum Message <> Reply to Message

On 8/30/12 10:51 AM, godber wrote:

> Is there a more succinct way of testing array membership than using where and n\_elements on

the indexes? Something like array\_contains(a, 'pancakes')?

> Austin

In general, how about:

print, where(a eq 'pancakes', /null) ? 'found' : 'not found'

Mike

-
Michael Galloy

www.michaelgalloy.com

Modern IDL: A Guide to IDL Programming (http://modernidl.idldev.com)

Research Mathematician

Tech-X Corporation

Subject: Re: Succinct way of testing array membership Posted by wlandsman on Thu, 30 Aug 2012 19:16:06 GMT

View Forum Message <> Reply to Message

For very large vectors, WHERE() methods are slow because (1) you compute the index of where the member is found, even if you just care about membership, and more important (2) WHERE operates on the entire vector, and does not exit as soon as the member is found. A faster method would be

function array\_contains,v,x; ls x an element of V? return,~array\_equal( (v eq x), 0b) end

since ARRAY\_EQUAL exits as soon as the first match is found. I can't imagine the speed difference is significant for normal size vectors, though.

--Wayne

On Thursday, August 30, 2012 12:51:48 PM UTC-4, godber wrote:

> Is there a more succinct way of testing array membership than using where and n\_elements on the indexes? Something like array\_contains(a, 'pancakes')?

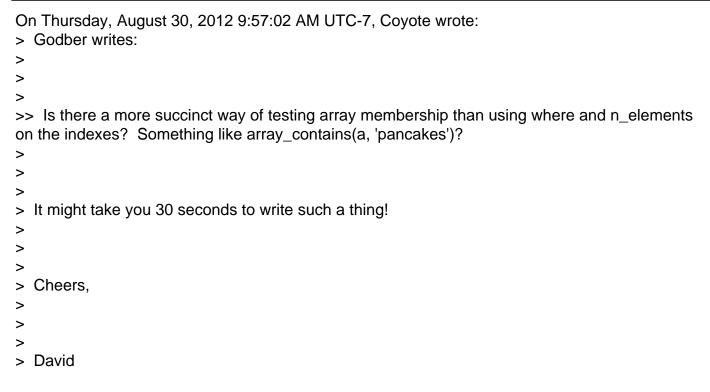
> >

>

> Austin

## Subject: Re: Succinct way of testing array membership Posted by godber on Thu, 30 Aug 2012 20:57:29 GMT

View Forum Message <> Reply to Message



Yeah David it does, but look at all the nice trys by people below. In fact some of them were very informative. I like Mike's suggestion, and it points out /null to me and wayne is kind enough to point out a quicker solution. Therefor I declare the question a success.

I mean, it still is a little surprising to me that there isn't an inbuilt way to test list or array membership.

Thanks for the alternatives guys.

Austin

>

Subject: Re: Succinct way of testing array membership Posted by Helder Marchetto on Thu, 30 Aug 2012 21:12:02 GMT View Forum Message <> Reply to Message

On Thursday, August 30, 2012 7:09:35 PM UTC+2, Mike Galloy wrote:

On 8/30/12 10:51 AM, godber wrote:
 Is there a more succinct way of testing array membership than using where and n\_elements on the indexes? Something like array\_contains(a, 'pancakes')?
 >>

```
>> Austin
>>
>
>
  In general, how about:
>
>
>
    print, where(a eq 'pancakes', /null) ? 'found' : 'not found'
>
>
>
>
  Mike
>
>
> Michael Galloy
  www.michaelgalloy.com
  Modern IDL: A Guide to IDL Programming (http://modernidl.idldev.com)
  Research Mathematician
> Tech-X Corporation
Hi Mike,
I never noticed the presence of the /null keyword. Nice tip.
However, the version you provided will not work (at least not on my pc...).
How about:
print, (where(a EQ 'pancakes', /null) NE !NULL) ? 'found' : 'not found'
print, ((where(a EQ 'pancakes'))[0] GE 0) ? 'found' : 'not found'
Not as clean, but still doing the job.
Cheers,
Helder
```