

---

Subject: Re: READ\_ASCII - accessing data from structures  
Posted by [David Fanning](#) on Wed, 12 May 2004 17:44:09 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Martin Doyle writes:

> I wonder if anyone might be able to help me with this?  
>  
> I'm using the IDL READ\_ASCII function to read in a semicolon seperated  
> file, after which I'd like to access individual elements of that data  
> file. From the RSI website, I see that I can access the data fields  
> (i.e. each column of data) by using, for example;  
>  
> print, mydata.(4)  
>  
> using the Variable\_Name.(Tag\_Index) method.  
>  
> Could someone tell me how to get to the individual elements of  
> mydata.(4)?? Using the above I can only get a large stream of numbers,  
> I need to use them one by one.

Well, this should work:

```
Print, (mydata.(4))[5]
```

But usually, you would just pull the column off and treat it as an array, unless you had some speciific memory considerations:

```
col4 = Reform(mydata.(4))  
print, col4[5]
```

Cheers,

David

--

David Fanning, Ph.D.  
Fanning Software Consulting  
Coyote's Guide to IDL Programming: <http://www.dfanning.com/>

---

---

Subject: Re: READ\_ASCII - accessing data from structures  
Posted by [JD Smith](#) on Wed, 12 May 2004 20:25:44 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

On Wed, 12 May 2004 11:44:09 -0600, David Fanning wrote:

```
> Martin Doyle writes:
>
>> I wonder if anyone might be able to help me with this?
>>
>> I'm using the IDL READ_ASCII function to read in a semicolon seperated
>> file, after which I'd like to access individual elements of that data
>> file. From the RSI website, I see that I can access the data fields
>> (i.e. each column of data) by using, for example;
>>
>> print, mydata.(4)
>>
>> using the Variable_Name.(Tag_Index) method.
>>
>> Could someone tell me how to get to the individual elements of
>> mydata.(4)?? Using the above I can only get a large stream of numbers,
>> I need to use them one by one.
>
> Well, this should work:
>
> Print, (mydata.(4))[5]
```

Just wanted to point out that `mydata[5].(4)` will typically be quicker if the `mydata` vector is quite long, and also requires fewer parentheses. Your method first creates a (potentially long) vector, and then subscripts it.

JD

---