Subject: get IDL to issue warning or throw error for mismatched arrays
Posted by Markus Schmassmann on Thu, 16 Jun 2016 14:59:00 GMT
View Forum Message <> Reply to Message

is there any way to get IDL to issue a warning or throw an error in for
mismatched arrays like those below?

indgen(2,2)+indgen(3,3)

Subject: Re: get IDL to issue warning or throw error for mismatched arrays
Posted by Dick Jackson on Thu, 16 Jun 2016 16:43:33 GMT
View Forum Message <> Reply to Message

On Thursday, 16 June 2016 07:59:02 UTC-7, Markus Schmassmann wrote:
> is there any way to get IDL to issue a warning or throw an error in for
> mismatched arrays like those below?
>
> indgen(2,2)+indgen(3,3)

Hi Markus,

Here is the behaviour we see, that the result is the size of the "overlapping" of the two arrays:

```
IDL> help,indgen(2,2)+indgen(3,3)
<Expression>    INT       = Array[2, 2]
```

I was hopeful that COMPILE_OPT STRICTARRSUBS would help. Here's what it does, before:

```
IDL> a=indgen(5)
IDL> a[[3,4,5,6]]
       3       4       4       4
```

... and after:

```
IDL> compile_opt strictarrsubs
IDL> a[[3,4,5,6]]
% Array used to subscript array contains out of range subscript: A.
% Execution halted at: $MAIN$
```

So, the STRICTARRSUBS is making a difference in that regard, but...

```
IDL> help,indgen(2,2)+indgen(3,3)
<Expression>    INT       = Array[2, 2]
```

It does not change the behaviour with different-shaped arrays. If you have to test arrays a and b for differing shape, a simple check is:

IF ~Array_Equal(Size(a, /DIMENSIONS), Size(b, /DIMENSIONS)) THEN ...

Hope this helps,
-Dick

Dick Jackson Software Consulting Inc.
Victoria, BC, Canada --- http://www.d-jackson.com

Subject: Re: get IDL to issue warning or throw error for mismatched arrays
Posted by Markus Schmassmann on Fri, 17 Jun 2016 12:50:53 GMT
View Forum Message <> Reply to Message

On 06/16/2016 06:43 PM, Dick Jackson wrote:
> On Thursday, 16 June 2016 07:59:02 UTC-7, Markus Schmassmann wrote:
>> is there any way to get IDL to issue a warning or throw an error in
>> for mismatched arrays like those below?
>>
>> indgen(2,2)+indgen(3,3)
> Here is the behaviour we see, that the result is the size of the
> "overlapping" of the two arrays: IDL> help,indgen(2,2)+indgen(3,3)
> <Expression> INT = Array[2, 2]
>
> I was hopeful that COMPILE_OPT STRICTARRSUBS would help. [...]
> but...
>
> IDL> help,indgen(2,2)+indgen(3,3) <Expression> INT = Array[2, 2]
>
> It does not change the behaviour with different-shaped arrays. If
> you have to test arrays a and b for differing shape, a simple check
> is:
>
> IF ~Array_Equal(Size(a, /DIMENSIONS), Size(b, /DIMENSIONS)) THEN ...
>
> Hope this helps,
unfortunately this is not news for me, I'm just looking for a way to
catch my own programming errors without many otherwise unnecessary tests

---

Subject: Re: get IDL to issue warning or throw error for mismatched arrays
Posted by wlandsman on Thu, 05 Jan 2017 03:25:49 GMT
View Forum Message <> Reply to Message

On Thursday, June 16, 2016 at 10:59:02 AM UTC-4, Markus Schmassmann wrote:
> is there any way to get IDL to issue a warning or throw an error in for
> mismatched arrays like those below?

>
> indgen(2,2)+indgen(3,3)

Just letting off steam here. I know that IDL does not have the ability to throw a warning for mismatched arrays. But I just embarrassed myself by reporting results based on a division y = a/b where a,b were both supposed to be 4096 x 4096 arrays. But I forgot to trim variable b from size 4100 x 4100 so I was dividing different sized arrays (and not getting any error from IDL).

I suppose there are times when one wants to divide different size arrays. I just can't think of any.

Subject: Re: get IDL to issue warning or throw error for mismatched arrays
Posted by Bill Nel on Thu, 05 Jan 2017 23:12:52 GMT
View Forum Message <> Reply to Message

On Thursday, June 16, 2016 at 10:59:02 AM UTC-4, Markus Schmassmann wrote:
> is there any way to get IDL to issue a warning or throw an error in for
> mismatched arrays like those below?
>
> indgen(2,2)+indgen(3,3)

I call the following routine, if there's any doubt about the size of the arrays I'm using:

```
;+
;*************************************************
FUNCTION ArrayTruncation, a, b, c, d
;*************************************************
;~ Detects whether array operations on a set of variables causes silent truncation. [Programming]
;
;  IDL silently truncates the results of array operations to the smaller of the
;  array operands. This is almost never what I want! It's particularly insidious
;  when one of the operands is a one-element array produced by something like where().
;
;  This routine detects such array truncation. It may be called with two, three or four arguments.
;
;  Examples
;  IDL> print, ArrayTruncation( [1,2,3], 4)
;    0
;  IDL> print, ArrayTruncation( [1,2,3], [4])
;    1
;  IDL> print, ArrayTruncation( [1,2,3], [4,5,6], 7)
;    0
;  IDL> print, ArrayTruncation( [1,2,3], [6], 7)
;    1
;  IDL> print, ArrayTruncation( [1,2,3], [6], 7, [4,5,6])
;    1
;
;  Notes;
```

```
;  1) The method is to check whether the sum of the arguments has the same number of elements as the
;     argument with the largest number of elements.
;
;  See also:
;-

;  Modification history:
;
;  Oct 30 2007  W. Rigby, original.
;  May 23 2008  changed oneliner category.
;  Mar 11 2009  changed oneliner.
;  Jul 27 2011  added a fourth argument
;  Sep 16 2016  tweaked error messages

compile_opt DEFINT32, STRICTARRSUBS, STRICTARR

On_Error, !my_on_error

MsgPrefix = "[ArrayTruncation] "

case n_params() of
   2: begin
      na = n_elements(a)
      nb = n_elements(b)
      if ((na EQ 0) || (nb EQ 0)) then Message, /noname, MsgPrefix + "One or more of the arguments is undefined"
      result = ( n_elements(a + b) NE max( [na, nb]))
   end

   3: begin
      na = n_elements(a)
      nb = n_elements(b)
      nc = n_elements(c)
      if ((na EQ 0) || (nb EQ 0) || (nc EQ 0)) then Message, /noname, MsgPrefix + "One or more of the arguments is undefined"
      result = ( n_elements(a + b + c) NE max( [na, nb, nc]))
   end

   4: begin
      na = n_elements(a)
      nb = n_elements(b)
      nc = n_elements(c)
      nd = n_elements(d)
      if ((na EQ 0) || (nb EQ 0) || (nc EQ 0) || (nd EQ 0)) then Message, /noname, MsgPrefix + "One or more of the arguments is undefined"
      result = ( n_elements(a + b + c + d) NE max( [na, nb, nc, nd]))
   end
```

```
   else: Message, /noname, "Usage: ArrayTruncation(a, b [c [d]])"
endcase

return, result

end
```

Subject: Re: get IDL to issue warning or throw error for mismatched arrays
Posted by lecacheux.alain on Fri, 06 Jan 2017 13:21:23 GMT
View Forum Message <> Reply to Message

Le jeudi 16 juin 2016 16:59:02 UTC+2, Markus Schmassmann a écrit :
> is there any way to get IDL to issue a warning or throw an error in for
> mismatched arrays like those below?
>
> indgen(2,2)+indgen(3,3)

You might use the list::ToArray() method, and throw the error :

```
IDL> help, (list(indgen(2,2),indgen(3,3))).ToArray()
% LIST::TOARRAY: Unable to concatenate arrays: Element 1
% Execution halted at: $MAIN$
```

while :

```
IDL> help, (list(indgen(2,2),indgen(2,2))).ToArray()
<Expression>    INT       = Array[2, 2, 2]
```

Cheers,
alx.
