
Subject: Re: Why doesn't this return the correct value?

Posted by [Spon](#) on Thu, 14 Feb 2008 11:22:53 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Feb 14, 10:59 am, chloesharro...@gmail.com wrote:

> Dear all

>

> I currently have an array called good_index. Some of the elements are ≥ 0 and others are < 0 .

I want to write a code that firstly finds how

>

> many of these elements are greater than zero and use this number to

> produce another array (called good_indexed_precip) with that number of

> elements. Then I want it to search through each element of good_index

> in turn. If upon searching it finds that the element is ≥ 0 I want it

> to find the value of that element in good_index. I then want to find

> the data stored in the index given by that value in another array

> called precip_change. The code I've written is below:

>

> =====

> counter_2 = 0

> number = total(good_index ≥ 0 , /int) ;this counts how many elements

> in good_index are ≥ 0

> print, number ;this gives the correct answer 6

>

> FOR s=0, (N_ELEMENTS(good_index)-1) DO BEGIN

> good_indexed_precip=fltarr(number)

> IF good_index[s] ≥ 0 THEN BEGIN

> good_indexed_precip[counter_2] = precip_change[good_index[s]]

> counter_2++

> ENDIF

> ENDFOR

> =====

>

> Unfortunately, it then prints good_indexed_precip with all 6 elements

> zero, even though the values of precip_change[good_index[s]] are non-

> zero.

>

> If I were to put in the data by hand, eg say

> good_indexed_precip[0] = precip_change[good_index[0]]

> good_indexed_precip[1] = precip_change[good_index[13]]

> This works fine, so I can't see why my loop isn't working correctly.

> I know that the 0th & 1st element of good_index are both ≥ 0 , yet if

> I run the loop: FOR s=0, 1 DO BEGIN etc and print good_indexed_precip

> it only has a non-zero value in the 1st element and not the 0th

> element!

>

> Thanks for all your help in advance.

> Chloé

```
>     FOR s=0, (N_ELEMENTS(good_index)-1) DO BEGIN
>         good_indexed_precip=fltarr(number)
There's the reason it's not working - you're redefining your array
every time. Take the second line out of the loop.
```

In terms of doing what you want to do, you should really look into using WHERE.

```
IndexOfIndices = WHERE(Good_index GT 0, Number)
Positive_Good_Index = Good_index[IndexOfIndices]
Good_Indexed_Precip = Precip_Change[Positive_Good_Index]
```

Take care,
Chris
