Subject: Avoiding redefinition of variable within loop.
Posted by kcwhite91 on Tue, 10 Jun 2014 15:43:06 GMT
View Forum Message <> Reply to Message

I have a 2-d array, with fixed first dimension. The second dimension changes length with each iteration of the loop through the first dimension. I want to compare each iteration to the one before it to find indices that disappeared between iterations, but because of the varying size I have to redefine it every time. I think there may be a way to avoid this using pointers and a structure, but I'm not sure how to do this.

```
for j=0, 18 do begin & $
l=where(zall[*,*,1,j] ne -32768.0) & $
k=where(zall1[l,1,j] gt 10) & $
index=intarr(19, n_elements(k)) & $
index[j, *]=l[k] & $
if (n_elements(k) gt 1) then begin & $
indv_echotop=fltarr(n_elements(k)) & $
      if (j ne 0) then begin & $
      for b=0, n_elements(index[j,*]-1) do begin & $
      x=where(index[j-1, *] eq index[j, b]) & $
      endfor & $
      endif & $
endif & $
endfor
```

index is my issue. I believe I need to define it as a structure with 19 different fields, one for each iteration.

Any help would be greatly appreciated.