
Subject: Re: converting int array to a string
Posted by [mole6e23](#) on Wed, 21 Nov 2001 16:57:09 GMT
[View Forum Message](#) <> [Reply to Message](#)

> function time2str, x

Chad -

I took the liberty of adding to your program a little. Notice that if you put in:

IDL> print, time2str([1,2,3])

you'd get back '0-59'. My revised function below fixes this and adds support for all ranges (with a min and max keyword so you can set your time2str function up). It also removes duplicate elements, and sorts the array. I also got rid of all the calls to strtrim, and replaced them with one call to strcompress at the end.

Todd

--

```
function time2str, x
  return, getnumberrange( x, min=0, max=59 )
end ;; time2str
```

```
function getnumberrange, x, min=min, max=max
  if( n_elements( x ) eq 0 ) then return, "
```

```
  tempX = x
  if( n_elements( min ) gt 0 ) then begin
    good = where(tempX ge min[0], ngood )
    if( ngood gt 0 ) then $
      tempX = tempX[good] $
    else return, "
  endif
```

```
  if( n_elements( max ) gt 0 ) then begin
    good = where(tempX le max[0], ngood )
    if( ngood gt 0 ) then $
      tempX = tempX[good] $
    else return, "
  endif
```

```
  newX = tempX[uniq( tempX, sort(tempX) )]
```

```
  ;; No unique elements, just return first element
```

```

if( n_elements( newX ) eq 1 ) then return, strtrim( tempX[0],2 )

;; See which elements in new X differ by only 1 (they will become
;; part of a range of numbers)
dx=(newX-shift(newX,1))[1:*)

;; Get number of elements where spread is greater than 1
;; (non-contiguous elements)
l=where(dx ne 1)

outStr=""
nl=n_elements(l)
nx=n_elements(newX)

if l[0] eq -1 then outStr=string( newX[0],'- ',newX[nx-1]) else begin
  if l[0] eq 0 then outStr=string(newX[0])+' ',
  if l[0] ne 0 then $
    outStr=string(newX[0])+'- '+string(newX[l[0]])+' ',
  for i=0, nl-2 do begin
    if ((l[i+1]-l[i]) gt 1) then begin
      outStr=outStr+string(newX[l[i]+1])+'- '+string(newX[l[i+1]])+ ' ',
    endif else begin
      outStr=outStr+string(newX[l[i+1]])+' ',
    endelse
  endfor
  if (nx-1)-l[(nl-1)] eq 1 then $
    outStr=outStr+string(newX[nx-1]) else $
    outStr=outStr+string(newX[(l[(nl-1))+1])+'- '+string(newX[nx-1 ])
endelse
return, strcompress(outStr, /remove_all)
end ;; getnumberrange

```
