Subject: Re: storing in array
Posted by kishore1818 on Mon, 12 May 2008 17:29:44 GMT
View Forum Message <> Reply to Message

On May 12, 1:06 pm, Spon <christoph.b...@gmail.com> wrote:
> On May 12, 4:35 pm, kishore1...@gmail.com wrote:
>
>
>
>> Hi,
>> Thanks for your suggestions, I modified according your suggestions.
>> Pl. go through my program
>> pro vfm_feature_flags1,val
>
> FUNCTION VFM_FEATURE_FLAGS1, Input
>
>
>
>> ; this routine demonstrates how to read and extract values from a
>> feature
>> ; classification flag 16-bit integer value in CALIPSO Level 2
>> Vertical
>> ; Feature Mask files
>> ;
>> ; INPUT:
>> ; val - the feature classification flag value to be decoded
>> ;
>> ; OUTPUT:
>> ; all information is printed into the IDL log window
>
> ; Make sure input is usable
> IF Input GT 65535L OR Input LT -32767L THEN $
> MESSAGE, 'Input not a 16-bit flag.'
>
> ; Convert input to 16-bit value
> Val = UINT(Input)
>
>
>
>> print, val
>
>> feature_type = 0
>> feature_type_qa = 0
>> ice_water_phase = 0
>> ice_water_phase_qa = 0
>> feature_subtype = 0
>> cloud_aerosol_psc_type_qa = 0
>> horizontal_averaging = 0

```
>
>> Output = { feature_type:"",                $
>>           feature_type_qa:"",              $
>>           ice_water_phase:"",              $
>>           ice_water_phase_qa:"",           $
>>           feature_subtype:"",                 $
>>           cloud_aerosol_psc_type_qa:"",     $
>>           horizontal_averaging:""}
>
>> for i=0,15 do begin
>>   if ((val and 2L^i) NE 0) then begin
>>     print,'Bit set: ',i+1
>>     case i+1 of
>>     1 : feature_type = feature_type + 1
>>     2 : feature_type = feature_type + 2
>>     3 : feature_type = feature_type + 4
>>     4 : feature_type_qa = feature_type_qa + 1
>>     5 : feature_type_qa = feature_type_qa + 2
>>     6 : ice_water_phase = ice_water_phase + 1
>>     7 : ice_water_phase = ice_water_phase + 2
>>     8 : ice_water_phase_qa = ice_water_phase_qa + 1
>>     9 : ice_water_phase_qa = ice_water_phase_qa + 2
>>     10 : feature_subtype = feature_subtype + 1
>>     11 : feature_subtype = feature_subtype + 2
>>     12 : feature_subtype = feature_subtype + 4
>>     13 : cloud_aerosol_psc_type_qa = cloud_aerosol_psc_type_qa + 1
>>     14 : horizontal_averaging = horizontal_averaging + 1
>>     15 : horizontal_averaging = horizontal_averaging + 2
>>     16: horizontal_averaging = horizontal_averaging + 4
>>     else:
>>     endcase
>>   endif
>> endfor
>
>> case feature_type of
>> 0 : Output.Feature_Type = 'invalid (bad or missing data)'
>> 1 : Output.Feature_Type = 'clear air''
>> 2 : begin
>>       Output.Feature_Type = 'cloud'
>>       case feature_subtype of
>>       0 : Output.Feature_Subtype = 'low overcast, transparent'
>>       1 : Output.Feature_Subtype = 'low overcast, opaque'
>>       2 : Output.Feature_Subtype = 'transition stratocumulus'
>>       3 : Output.Feature_Subtype = 'low, broken cumulus'
>>       4 : Output.Feature_Subtype = 'altocumulus (transparent)'
>>       5 : Output.Feature_Subtype = 'altostratus (opaque)'
>>       6 : Output.Feature_Subtype = 'cirrus (transparent)'
>>       7 : Output.Feature_Subtype = 'deep convective (opaque)'
```

```
>>       else : Output,'*** error getting Feature Subtype'
>
>  ELSE : Output.Feature_Subtype = '*** error getting Feature Subtype'
>
>>       endcase
>>     end
>>  3 : begin
>>       Output.Feature_Type = 'aerosol'
>>       case feature_subtype of
>>       0 : Output.Feature_Subtype = 'not determined'
>>       1 : Output.Feature_Subtype = 'clean marine'
>>       2 : Output.Feature_Subtype = 'dust'
>>       3 : Output.Feature_Subtype = 'polluted continental'
>>       4 : Output.Feature_Subtype = 'clean continental'
>>       5 : Output.Feature_Subtype = 'polluted dust'
>>       6 : Output.Feature_Subtype = 'smoke'
>>       7 : Output.Feature_Subtype = 'other''
>>       else : Output,'*** error getting Feature Subtype'
>
>  ELSE : Output.Feature_Subtype = '*** error getting Feature Subtype'
>
>>       endcase
>>     end
>>  4 : begin
>>       Output.Feature_Type = 'stratospheric feature--PSC or
>>  stratospheric aerosol'
>>       case feature_subtype of
>>       0 : Output.Feature_Subtype = 'not determined'
>>       1 : Output.Feature_Subtype = 'non-depolarizing PSC'
>>       2 : Output.Feature_Subtype = 'depolarizing PSC'
>>       3 : Output.Feature_Subtype = 'non-depolarizing aerosol'
>>       4 : Output.Feature_Subtype = 'depolarizing aerosol'
>>       5 : Output.Feature_Subtype = 'spare'
>>       6 : Output.Feature_Subtype = 'spare'
>>       7 : Output.Feature_Subtype = 'other'
>>       else : Output,'*** error getting Feature Subtype'
>
>  ELSE : Output.Feature_Subtype = '*** error getting Feature Subtype'
>
>>       endcase
>>     end
>>  5 : Output.Feature_Type = 'surface'
>>  6 : Output.Feature_Type = 'subsurface'
>>  7 : Output.Feature_Type = 'no signal (totally attenuated)'
>>  else : Output,'*** error getting Feature Type'
>
>  ELSE : Output.Feature_Type = '*** error getting Feature Type'
>
```

```
>>  endcase
>
>>  case feature_type_qa of
>>  0 : Output.Feature_Type_QA = 'none'
>>  1 : Output.Feature_Type_QA = 'low'
>>  2 : Output.Feature_Type_QA = 'medium'
>>  3 : Output.Feature_Type_QA = 'high'
>>  else : Output,'*** error getting Feature Type QA'
>
>  ELSE : Output.Feature_Type_QA = '*** error getting Feature Type QA'
>
>>  endcase
>
>>  case ice_water_phase of
>>  0 : Output.Ice_Water_Phase = 'unknown/not determined'
>>  1 : Output.Ice_Water_Phase = 'ice'
>>  2 : Output.Ice_Water_Phase = 'water'
>>  3 : Output.Ice_Water_Phase = 'mixed phase'
>>  else : Output,'*** error getting Ice/Water Phase'
>
>  ELSE : Output.Ice_Water_Phase = '*** error getting Ice/Water Phase'
>
>>  endcase
>
>>  case ice_water_phase_qa of
>>  0 : Output.Ice_Water_Phase_QA = 'none'
>>  1 : Output.Ice_Water_Phase_QA = 'low'
>>  2 : Output.Ice_Water_Phase_QA = 'medium'
>>  3 : Output.Ice_Water_Phase_QA = 'high'
>>  else : Output,'*** error getting Ice/Water Phase QA'
>
>  ELSE : Output.Ice_Water_Phase_QA = '*** error getting Ice/Water Phase
>  QA'
>
>
>
>>  endcase
>
>>  if (cloud_aerosol_psc_type_qa eq 0) then begin
>>    Output.Cloud_Aerosol_PSC_Type_QA = 'not confident'
>>  endif else begin
>>    Output.Cloud_Aerosol_PSC_Type_QA = 'confident'
>>  endelse
>
>>  case horizontal_averaging of
>>  0 : Output.Horizontal_averaging = 'not applicable'
>>  1 : Output.Horizontal_averaging = '1/3 km'
>>  2 : Output.Horizontal_averaging = '1 km'
```

>> 3 : Output.Horizontal_averaging = '5 km'
>> 4 : Output.Horizontal_averaging = '20 km'
>> 5 : Output.Horizontal_averaging = '80 km'
>> else : Output,'*** error getting Horizontal averaging'
>
> ELSE : Output.Ice_Water_Phase = '*** error getting Ice/Water Phase'
>
>
>
>
>
>> endcase
>> help,Output,/structure
>> ;Return, Output
> RETURN, Output
>> end
>> Using above routine I wrote a small progam
>> data1=[44474, 36282, 28602, 28090]
>
>> for i=0,3 do begin
>>   ;vfm_feature_flags,data1(i)
>>   vfm_feature_flags1,data1(i)
>>   pause='' & read,pause
>
>> endfor
>> end
>
>> Only, problem is how to recall output parameters.
>
> Replace this very first line:
> pro vfm_feature_flags,val
>
> with
>
> function vfm_feature_flags,val
>
> then, down at the very end of the programme, remove the comment-out
> from this line
>   ;Return, Output
>
> IDL has two types of programmes, 'procedures' and 'functions'.
> Functions will return an output. Procedures can only alter input
> parameters (although this can be used to effectively give output too).
> Functions are called slightly differently to procedures, though the
> code language in them is identical apart from the 'Return, A_Value'
> thing.
>
> To call the function, use:

> result = vfm_feature_flags(value)
>
> e.g.
> myoutput = vfm_feature_flags(36282u)
>
> Then you can look at the content of the output like this:
> help, myoutput, /structure
>
> or something like this:
> print, myoutput.feature_type
> to get a single output field from the structure.
>
> Regards,
> Chris

Hi Chris,

Now, its working fine, thanks a lot for your kind help.
This forum is very good for new IDL guys.

Regards,

Kishore
