d2scomp

integer d2scomp(integer *quality*, integer *baseline*)

quality	IN:	JPEG quality specification
baseline	IN:	JPEG baseline specification

Purpose	Fortran-specific routine that sets the parameters needed for the JPEG algorithm.
Return value	Returns `SUCCEED` (or `0`) if successful and `FAIL` (or `-1`) otherwise.
Description	**d8sjpeg** changes the JPEG compression parameter settings set in the **d8scomp** routine.

d2sjpeg

integer d2sjpeg(integer *quality*, integer *baseline*)

quality	IN:	JPEG quality specification
baseline	IN:	JPEG baseline specification

Purpose	Fortran-specific routine that sets the parameters needed for the JPEG algorithm.
Return value	Returns `SUCCEED` (or `0`) if successful and `FAIL` (or `-1`) otherwise.
Description	**d2sjpeg** changes the JPEG compression parameter settings set in the **d2scomp** routine.

d8scomp

integer d8scomp(integer *quality*, integer *baseline*)

quality	IN:	JPEG quality specification
baseline	IN:	JPEG baseline specification

Purpose Fortran-specific routine that sets the parameters needed for the JPEG algorithm.

Return value Returns `SUCCEED` (or `0`) if successful and `FAIL` (or `-1`) otherwise.

Description **d8sjpeg** changes the JPEG compression parameter settings set in the **d8scomp** routine.

d8sjpeg

integer d8sjpeg(integer *quality*, integer *baseline*)

quality	IN:	JPEG quality specification
baseline	IN:	JPEG baseline specification

Purpose Fortran-specific routine that sets the parameters needed for the JPEG algorithm.

Return value Returns `SUCCEED` (or `0`) if successful and `FAIL` (or `-1`) otherwise.

Description **d8sjpeg** changes the JPEG compression parameter settings set in the **d8scomp** routine.

DFgetcomp

int DFgetcomp(int32 *file_id*, uint16 *tag*, uint16 *ref*, uint8 **image*, int32 *xdim*, int32 *ydim*, int16 *method*)

file_id	IN:	Name of the file
tag	IN:	Tag of the image to be compressed
ref	IN:	Reference number of the image to be compressed
image	IN	Pointer to the memory block the image will be stored in
xdim	IN	Length of the x dimension of the compressed image
ydim	IN	Length of the y dimension of the compressed image
method	IN	Compression method to be used

Purpose Compresses and writes image data to the specified file.

Return value Returns `SUCCEED` (or `0`) if successful and `FAIL` (or `-1`) otherwise.

Description The IMCOMP compression method modifies the palette associated with the image.. This is a general compression interface - to be used anytime image compression is needed in HDF.

The space needed for compression and decompression can be allocated statically or dynamically, depending on the `DF_DYNAMIC` flag. It can be allocated for the entire image or in part depending on memory availability. Accordingly, writing out the image can be done in its entriety or by row.

Note that run-length encoding (or RLE) compression is always by row.

DFputcomp

intn DFputcomp(int32 *file_id*, uint16 *tag*, uint16 *ref*, uint8 **image*, int32 *xdim*, int32 *ydim*, uint8 **palette*, uint8 **newpal*, int16 *method*, comp_info **c_into*)

file_id	IN:	Name of the file
tag	IN:	Tag of the image to be compressed
ref	IN:	Reference number of the image to be compressed
image	IN	Pointer to the memory block the image will be stored in
xdim	IN	Length of the x dimension of the compressed image
ydim	IN	Length of the y dimension of the compressed image
palette	IN	Pointer to the palette associated with the image
newpal	IN	Pointer to a modified palette- produced with the IMCOMP compression method
method	IN	Number of rows in the image
c_info	IN	Number of rows in the image

Purpose Writes a 24-bit image to the specified file.

Return value Returns `SUCCEED` (or `0`) if successful and `FAIL` (or `-1`) otherwise.

Description **DF24addimage** appends a 24-bit raster image set for the image to the file. Array *image* is assumed to be width *x* height *x* 3 bytes. In Fortran-77, the dimensions of the array *image* must be the same as the dimensions of the image data.

The order in which dimensions are declared is different between C and Fortran-77. Ordering varies because Fortran-77 arrays are stored in column-major order, while C arrays are stored in row-major order. (Row-major order implies that the last coordinate varies fastest).