Preliminary Test Procedure for IRAF
IRAF Version V2.11

Jeannette Barnes
Central Computer Services
National Optical Astronomy Observatories
P.O. Box 26732
Tucson, AZ 85726

Revised September 23, 1997

The following pages describe a short test procedure that sites can execute to test some basic image functions within IRAF for a new installation. This process will help verify that everything is working correctly and also help the first time user gain familiarity with the system. The commands you need to type and the expected terminal output are given below.

We will assume that you have started IRAF and are residing in an empty directory from which you wish to work.

1. All of the IRAF core packages are loaded when you log into IRAF. You can list what packages are currently loaded by typing the word `package`. The following should be displayed, but the packages may not be listed in the same order.

```
cl> package
    clpackage
    language
    utilities
    proto
    tv
    dataio
    plot
    imutil
    immatch
    imgeom
    imfit
    imfilter
    imcoords
    images
    user
```

```
    system
    lists
    noao
```

New packages can be loaded by simply typing the package name. Note the change to the prompt. The last package loaded can be unloaded by typing `bye`. Try the following. Note that in our example the top level packages listed may be different than yours.

```
cl> digiphot
      apphot.   daophot.  photcal.  ptools.
di> bye
      ctio.       images.     nlocal.     plot.       system.
      dataio.     iue.        noao.       proto.      utilities.
      dbms.       language.   obsolete.   softools.   vol.
      grasp.      lists.      pipeline.   stsdas.
```

For this test we want to use all the default parameters for the various tasks. Thus we will `unlearn` each of the packages that we will use. Note that this is not the usual procedure - generally, you want IRAF to "learn" or remember the parameters that were used previously. To do this, type

```
cl> unlearn dataio
cl> unlearn plot
cl> unlearn lists
cl> unlearn imcoords imfilter imfit imgeom immatch imutil tv
```

2. An IRAF image exists in the "dev" directory. Let's first make a copy of this image, putting the header file into the current working directory and the pixel files into your default image directory.

```
cl> imcopy dev$pix image.short
dev$pix -> image.short
```

3. Let's look at the header information for this image.

```
cl> imhead image.short long+ | page
```

should produce a listing similar to the following:

```
image.short[512,512][short]: m51  B  600s
    No bad pixels, no histogram, min=unknown, max=unknown
    Line storage mode, physdim [512,512], length of user area 1540 s.u.
    Created Mon 09:31:01 11-May-92, Last modified Mon 09:31:02 11-May-92
    Pixel file 'gemini!/scr3/iraf/jbarnes/image.short.pix' [ok]
```

```
'KPNO-IRAF'           /
'08-04-92'            /
New copy of dev$pix
IRAF-MAX=           1.993600E4  /  DATA MAX
IRAF-MIN=          -1.000000E0  /  DATA MIN
IRAF-BPX=                   16  /  DATA BITS/PIXEL
IRAFTYPE= 'SHORT   '            /  PIXEL TYPE
CCDPICNO=                   53  /  ORIGINAL CCD PICTURE NUMBER
ITIME   =                  600  /  REQUESTED INTEGRATION TIME (SECS)
TTIME   =                  600  /  TOTAL ELAPSED TIME (SECS)
OTIME   =                  600  /  ACTUAL INTEGRATION TIME (SECS)
DATA-TYP= 'OBJECT (0)'          /  OBJECT,DARK,BIAS,ETC.
DATE-OBS= '05/04/87'            /  DATE DD/MM/YY
RA      = '13:29:24.00'         /  RIGHT ASCENSION
DEC     = '47:15:34.00'         /  DECLINATION
EPOCH   =                 0.00  /  EPOCH OF RA AND DEC
ZD      = '22:14:00.00'         /  ZENITH DISTANCE
UT      = ' 9:27:27.00'         /  UNIVERSAL TIME
ST      = '14:53:42.00'         /  SIDEREAL TIME
CAM-ID  =                    1  /  CAMERA HEAD ID
CAM-TEMP=              -106.22  /  CAMERA TEMPERATURE, DEG C
DEW-TEMP=              -180.95  /  DEWAR TEMPRATURE, DEG C
F1POS   =                    2  /  FILTER BOLT I POSITION
F2POS   =                    0  /  FILTER BOLT II POSITION
TVFILT  =                    0  /  TV FILTER
CMP-LAMP=                    0  /  COMPARISON LAMP
TILT-POS=                    0  /  TILT POSITION
BIAS-PIX=                    0  /
BI-FLAG =                    0  /  BIAS SUBTRACT FLAG
BP-FLAG =                    0  /  BAD PIXEL FLAG
CR-FLAG =                    0  /  BAD PIXEL FLAG
DK-FLAG =                    0  /  DARK SUBTRACT FLAG
FR-FLAG =                    0  /  FRINGE FLAG
FR-SCALE=                 0.00  /  FRINGE SCALING PARAMETER
TRIM    = 'Apr 22 14:11 Trim image section is [3:510,3:510]'
BT-FLAG = 'Apr 22 14:11 Overscan correction strip is [515:544,3:510]'
FF-FLAG = 'Apr 22 14:11 Flat field image is Flat1.imh with scale=183.9447'
CCDPROC = 'Apr 22 14:11 CCD processing done'
AIRMASS =    1.08015632629395   / AIRMASS
HISTORY 'KPNO-IRAF'
HISTORY '24-04-87'
```

Note that the pixels are short integers (=16 bits). Notice also that your pixel file path (line 5) is different from the one above - that, of course, is to be expected. (On VMS systems, the "." in the pixel file path name will be converted to "j7" to produce a legal VMS file name.)

It would be useful to generate two more copies of this image but with different pixel

types - one with 32-bit floating point pixels (called reals) and one with 64-bit double precision floating point pixels (called double). Note that IRAF also supports other pixel data types - 32-bit integers called long, 16-bit unsigned integers called ushort, and complex numbers. Execute the following:

```
cl> imarith image.short  /  1 image.real pixtype=r
cl> imarith image.short  /  1 image.dbl pixtype=d
cl> imhead image.*
image.dbl.imh[512,512][double]: m51  B  600s
image.real.imh[512,512][real]: m51  B  600s
image.short.imh[512,512][short]: m51  B  600s
```

Note the ".imh" extension to the image name - this declares that the image was written as an IRAF image rather than some other format. In many cases this extension is transparent to the user.

4. Let's execute a couple of more tasks that will exercise some image operators. Typing

```
cl> minmax image.dbl,image.real,image.short
    image.dbl [77,4] -1. [348,189] 19936.
    image.real  -1.  19936.
    image.short  -1.  19936.
```

Now execute

```
cl> listpix image.short[300:310,200:205] formats="%4s %4s" | table
```

The following table of pixel values should be displayed.

```
  1.   1.  145.     7.   2.  130.     2.   4.  149.     8.   5.  140.
  2.   1.  143.     8.   2.  132.     3.   4.  146.     9.   5.  143.
  3.   1.  141.     9.   2.  128.     4.   4.  143.    10.   5.  131.
  4.   1.  142.    10.   2.  132.     5.   4.  145.    11.   5.  148.
  5.   1.  135.    11.   2.  139.     6.   4.  140.     1.   6.  138.
  6.   1.  138.     1.   3.  162.     7.   4.  133.     2.   6.  139.
  7.   1.  134.     2.   3.  145.     8.   4.  129.     3.   6.  145.
  8.   1.  125.     3.   3.  146.     9.   4.  128.     4.   6.  141.
  9.   1.  130.     4.   3.  144.    10.   4.  141.     5.   6.  141.
 10.   1.  123.     5.   3.  135.    11.   4.  137.     6.   6.  149.
 11.   1.  132.     6.   3.  141.     1.   5.  144.     7.   6.  149.
  1.   2.  147.     7.   3.  127.     2.   5.  145.     8.   6.  147.
  2.   2.  147.     8.   3.  129.     3.   5.  133.     9.   6.  144.
  3.   2.  145.     9.   3.  131.     4.   5.  144.    10.   6.  143.
  4.   2.  141.    10.   3.  133.     5.   5.  145.    11.   6.  151.
  5.   2.  132.    11.   3.  135.     6.   5.  144.
  6.   2.  130.     1.   4.  149.     7.   5.  143.
```

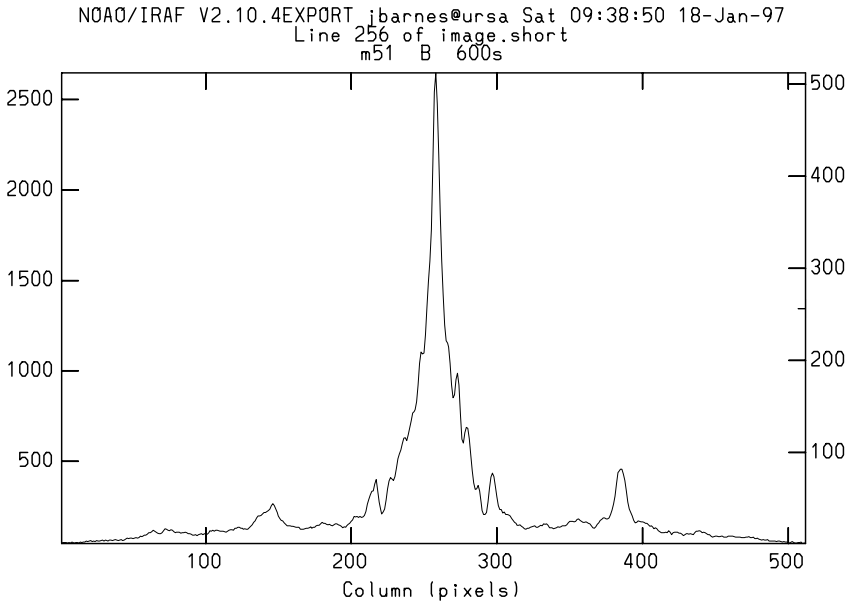

Figure 1: Line plot of image.short.

Similarly, the following executions of "listpixels" should generate the same table as above.

```
cl> listpix image.real[300:310,200:205] formats="%4s %4s" | table
cl> listpix image.dbl[300:310,200:205] formats="%4s %4s" | table
```

5. Now let's check some plotting options. Type

   ```
   implot image.short
   ```

   A plot similar to Figure 1 should be displayed - note that "implot" defaults to the center line (row) of the image.

   While the cursor is being displayed type

   ```
   :c 275
   ```

   and a plot similar to Figure 2 should be displayed.

   While still in "implot" and with the cursors displayed, executing the following set of commands should display the same plots for the other images.

   ```
   :i image.real
   :l 256
   :c 275
   :i image.dbl
   ```

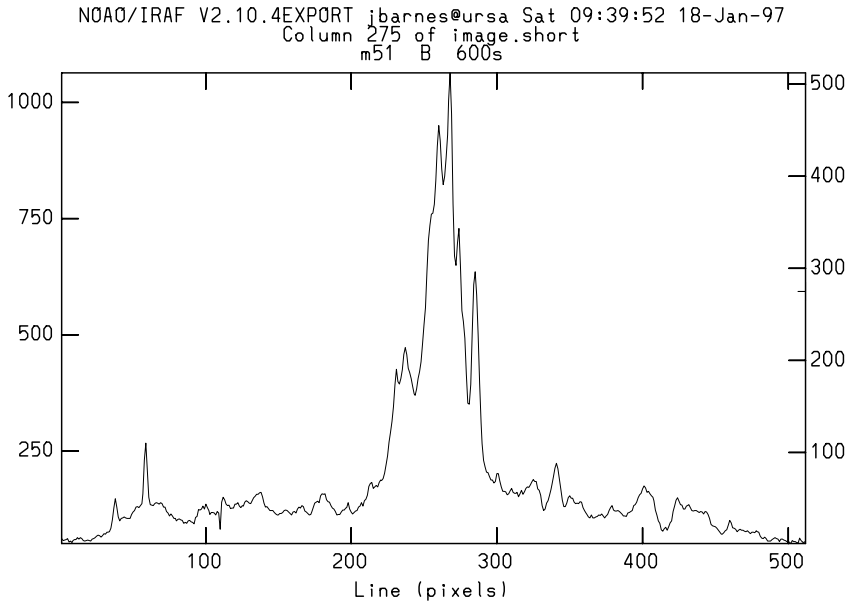

Figure 2: Column plot of image.short

```
:l 256
:c 275
```

There is another image in the "dev" directory (actually the same image but with a header that contains the RA and DEC for the picture). Let's switch to that data. Executing the following commands should produce the plot in Figure 3 - note the units on the x axis - RA instead of pixels.

```
:i dev$wpix
:l 240 245
:w world
:f %H
```

Note that if you lose the plot for some reason while you are in "implot" that you can get the plot back with `0`, the "zero". You can also generate a list of the cursor options available in "implot" by typing `?` while you are in cursor mode - the "return" key will get the plot back. Exit "implot" with `q`. Type `clear` to clear the screen if necessary.

6. Now let's test the use of image sections. Type and observe the following terminal interactions.

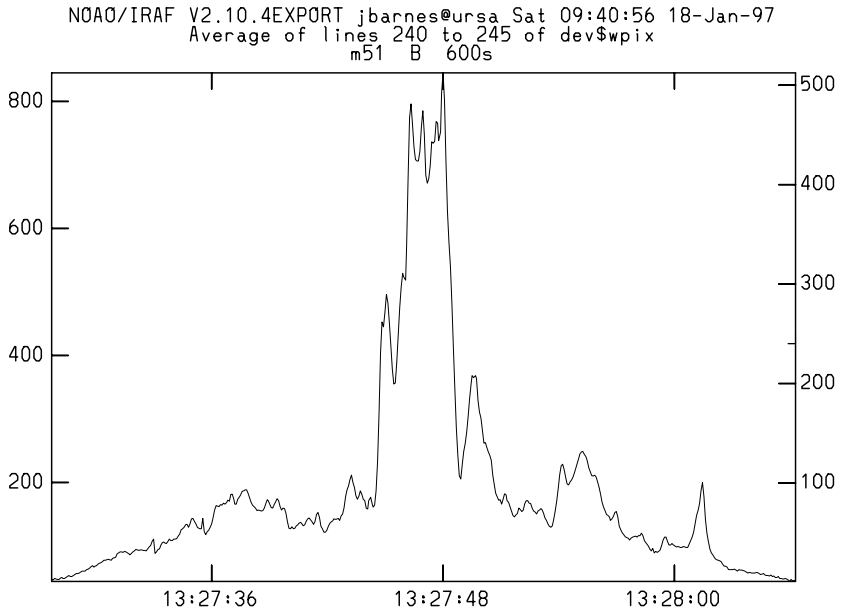

Figure 3: Line plot of dev$wpix showing world coordinate information.

```
cl> imcopy image.real[200:300,200:300] image.sect
image.real[200:300,200:300] -> image.sect
cl> imhead image.sect
image.sect[101,101][real]: m51  B  600s
```

7. Let's make a contour plot of this new image.

```
cl> contour image.sect
Image will be block averaged by 1 in x and 1 in y
```

A plot similar to Figure 4 should be displayed. The contour levels may be marked differently.

Let's again check some of the basic cursor plot options available in all of the plotting packages. Now type `=gcur`.

The plot and cursors should appear on your screen. Now move the cursors so that they intersect on the contour marked at x=60 and y=60. You can check to see if you have the right feature by typing `C` - that's a capital! Once you are set on the right feature type `Z`, again a capital, and the following "zoomed" plot should be displayed. Use `A` to put the proper axes on the plot. Note that you can generate a listing of the cursor options by typing `:.help` while you are in cursor mode - these cursor options are basic to most plotting tasks in IRAF. Exit the "cursor help mode" with `q` followed by a "return" to

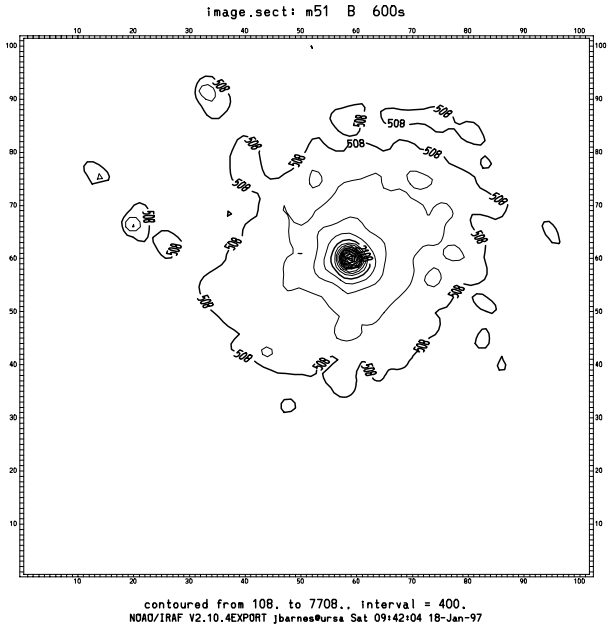

Figure 4: Contour plot of image.sect.

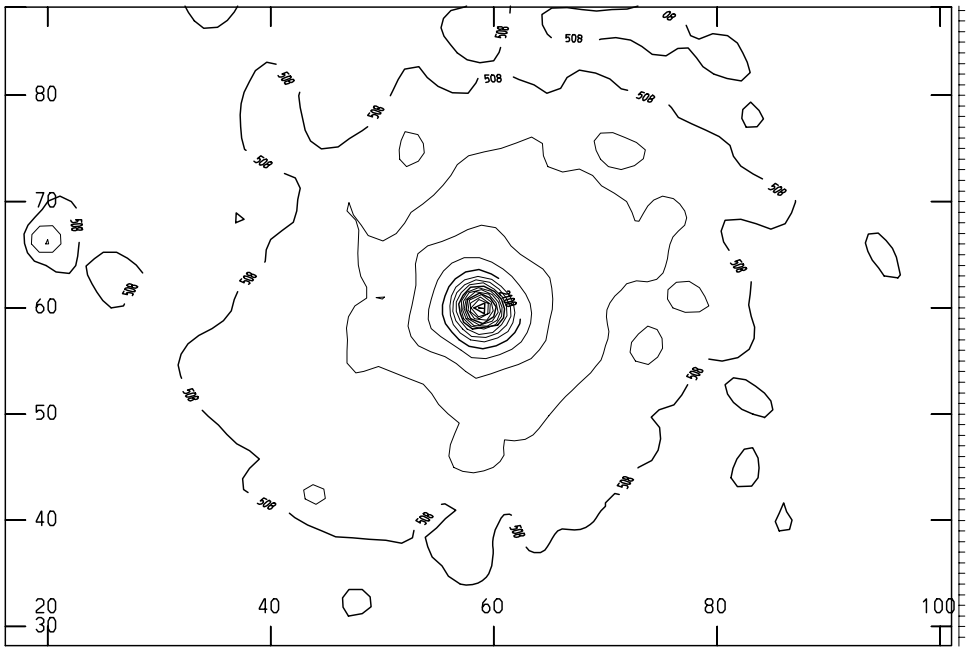

Figure 5: Zoomed plot.

replot. A `q` or any small letter keystroke will exit `=gcur`. Type `clear` to clear the screen if necessary.

8. If you are running a display server such as IMTOOL or SAOimage then we can check to be sure the image display facilities are working correctly. The following commands should produce the image in Figure 6. It may be necessary to increase the window size to fully see the coordinate grid labels.

```
cl> set stdimage=imt800
cl> display dev$wpix
frame to be written into (1:4) (1): 1
z1 = 36., z2 = 320.0713
cl> wcslab dev$wpix 1
```

9. At this time, let's modify a couple of image titles. Run the task "hedit" as below.

```
cl> hedit
images to be edited: image.real
fields to be edited: title
value expression: m51 real
image.real,i_title ("m51  B  600s" -> "m51 real"):
image.real,i_title: "m51  B  600s" -> "m51 real"
update image.real ? (yes):
image.real updated
cl> hedit
images to be edited (image.real): image.dbl
fields to be edited (title):
value expression (m51 real): m51 double
image.dbl,i_title ("m51  B  600s" -> "m51 double"):
image.dbl,i_title: "m51  B  600s" -> "m51 double"
update image.dbl ? (yes):
image.dbl updated
```

We can verify the new title with the "imheader" task.

```
cl> imhead image*.imh
image.dbl.imh[512,512][double]: m51 double
image.real.imh[512,512][real]: m51 real
image.sect.imh[101,101][real]: m51  B  600s
image.short.imh[512,512][short]: m51  B  600s
```

10. We will now concentrate on writing and reading images to/from tape in FITS format. You may want to physically mount your tape on the drive first if you are using a VMS

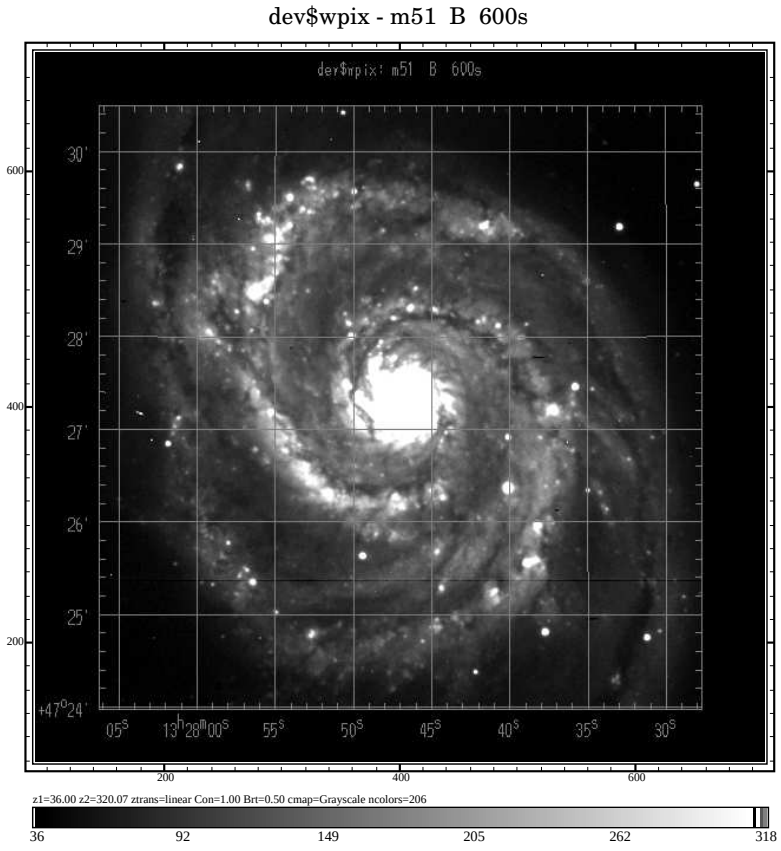

Figure 6: World coordinate grid overlaying dev$wpix image.

system since the IRAF "allocate" command does a "MOUNT/FOR" in VMS. Note also that IRAF refers to the tape devices as "mta", etc. So, please replace the proper drive notation in the following.

```
allocate mta
cl> wfits image.short mta new+
File 1: image.short -> mta[1] m51  B  600s          Size = 512 x 512
        pixtype=short bitpix=16 blkfac=10 scaling=none
        2 Header  183  Data logical (2880 byte) records written
```

11. Let's make a quick check of the tape writing.

```
cl> mtexamine mta
File mta[1]:
        18 28800-byte records
        1 14400-byte records
        Total 19 records, 532800 bytes
File mta[2]:
        Total 0 records, 0 bytes
Tape at EOT
cl> devstatus mta
# Magtape unit mta status Tue 08:23:42 12-May-92 user jbarnes
file = 2 (EOT)
record = 1
nfiles = 1
tapeused = 520
pflags = 0
```

12. Let's read the FITS image on the tape back onto disk as an IRAF image and check it.

    ```
    rfits mta 1 new.short long+ | page
    ```

    As the image is read in the FITS header should be displayed on your terminal.

    ```
    File: new.short
    SIMPLE  =                    T  /  FITS STANDARD
    BITPIX  =                   16  /  FITS BITS/PIXEL
    NAXIS   =                    2  /  NUMBER OF AXES
    NAXIS1  =                  512  /
    NAXIS2  =                  512  /
    BSCALE  =       1.0000000000E0  /  REAL = TAPE*BSCALE + BZERO
    BZERO   =       0.0000000000E0  /
    OBJECT  = 'm51  B  600s'        /
    ORIGIN  = 'KPNO-IRAF'           /
    DATE    = '12-05-92'            /
    IRAFNAME= 'image.short'         /  NAME OF IRAF IMAGE FILE
    IRAF-MAX=           1.993600E4  /  DATA MAX
    IRAF-MIN=          -1.000000E0  /  DATA MIN
    IRAF-BPX=                   16  /  DATA BITS/PIXEL
    IRAFTYPE= 'SHORT   '            /  PIXEL TYPE
    CCDPICNO=                   53  /  ORIGINAL CCD PICTURE NUMBER
    ITIME   =                  600  /  REQUESTED INTEGRATION TIME (SECS)
    TTIME   =                  600  /  TOTAL ELAPSED TIME (SECS)
    OTIME   =                  600  /  ACTUAL INTEGRATION TIME (SECS)
    DATA-TYP= 'OBJECT (0)'          /  OBJECT,DARK,BIAS,ETC.
    DATE-OBS= '05/04/87'            /  DATE DD/MM/YY
    RA      = '13:29:24.00'         /  RIGHT ASCENSION
    DEC     = '47:15:34.00'         /  DECLINATION
    ```

```
EPOCH   =                 0.00  /  EPOCH OF RA AND DEC
ZD      = '22:14:00.00'         /  ZENITH DISTANCE
UT      = ' 9:27:27.00'         /  UNIVERSAL TIME
ST      = '14:53:42.00'         /  SIDEREAL TIME
CAM-ID  =                    1  /  CAMERA HEAD ID
CAM-TEMP=              -106.22  /  CAMERA TEMPERATURE, DEG C
DEW-TEMP=              -180.95  /  DEWAR TEMPRATURE, DEG C
F1POS   =                    2  /  FILTER BOLT I POSITION
F2POS   =                    0  /  FILTER BOLT II POSITION
TVFILT  =                    0  /  TV FILTER
CMP-LAMP=                    0  /  COMPARISON LAMP
TILT-POS=                    0  /  TILT POSITION
BIAS-PIX=                    0  /
BI-FLAG =                    0  /  BIAS SUBTRACT FLAG
BP-FLAG =                    0  /  BAD PIXEL FLAG
CR-FLAG =                    0  /  BAD PIXEL FLAG
DK-FLAG =                    0  /  DARK SUBTRACT FLAG
FR-FLAG =                    0  /  FRINGE FLAG
FR-SCALE=                 0.00  /  FRINGE SCALING PARAMETER
TRIM    = 'Apr 22 14:11 Trim image section is [3:510,3:510]'
BT-FLAG = 'Apr 22 14:11 Overscan correction strip is [515:544,3:510]'
FF-FLAG = 'Apr 22 14:11 Flat field image is Flat1.imh with scale=183.9447'
CCDPROC = 'Apr 22 14:11 CCD processing done'
AIRMASS =    1.08015632629395   / AIRMASS
HISTORY 'KPNO-IRAF'
HISTORY '24-04-87'
HISTORY 'KPNO-IRAF'           /
HISTORY '08-04-92'            /
HISTORY New copy of dev$pix
END
    No FITS extensions Image renamed to new.short
```

13. Let's check the image now.

```
cl> imarith image.short / new.short div.short
cl> minmax div.short
    div.short [1,1] 1. [1,1] 1.
```

14. Something new! Let's make an "@file".

```
cl> files image.r*.imh > fitsout
cl> files image.d*.imh >> fitsout
cl> type fitsout
image.real.imh
image.dbl.imh
```

15. Using the file that we just created, let's add those images to our tape and then examine the tape.

```
cl> wfits @fitsout mta new- bit=16
File 1: image.real.imh -> mta[EOT] m51 real              Size = 512 x 512
        pixtype=real bitpix=16 blkfac=10 bscale=0.3042241 bzero=9967.5
        2 Header  183  Data logical (2880 byte) records written
File 2: image.dbl.imh -> mta[EOT] m51 double            Size = 512 x 512
        pixtype=double bitpix=16 blkfac=10 bscale=0.3042241 bzero=9967.5
        2 Header  183  Data logical (2880 byte) records written
cl> mtex mta
File mta[1]:
        18 28800-byte records
        1 14400-byte records
        Total 19 records, 532800 bytes
File mta[2]:
        18 28800-byte records
        1 14400-byte records
        Total 19 records, 532800 bytes
File mta[3]:
        18 28800-byte records
        1 14400-byte records
        Total 19 records, 532800 bytes
File mta[4]:
        Total 0 records, 0 bytes
Tape at EOT
cl> devstatus mta
# Magtape unit mta status Tue 08:23:42 12-May-92 user jbarnes
file = 4 (EOT)
record = 1
nfiles = 3
tapeused = 1560
pflags = 0
```

16. Let's make one more check on "rfits".

```
cl> rfits mta 1,3 check
File:  mtc[1]   -> check00010000.imh  m51  B  600s          size=512x512
    bitpix=16  scaling=none  pixtype=short
    No FITS extensions Image renamed to check0001
File:  mtc[3]   -> check00030000.imh  m51 double            size=512x512
    bitpix=16  bscale=0.3042888  bzero=9967.5  pixtype=real
    No FITS extensions Image renamed to check0003
```

```
cl> imhead check*.imh
check0001.imh[512,512][short]: m51  B  600s
check0003.imh[512,512][real]: m51 double
```

Note that "rfits" chose an appropriate datatype on input for us. We could have selected another datatype by modifying the parameters for "rfits". As you probably remember, the first image on the tape is image.short and the third image on the tape is image.dbl.

The "imstatistics" task should help verify our results.

```
cl> imstat *.imh
#               IMAGE      NPIX      MEAN    STDDEV       MIN       MAX
        check0001.imh    262144     108.3     131.3      -1.    19936.
        check0003.imh    262144     108.5     131.3   -0.7055    19936.
        div.short.imh    262144        1.        0.        1.        1.
        image.dbl.imh    262144     108.3     131.3      -1.    19936.
       image.real.imh    262144     108.3     131.3      -1.    19936.
       image.sect.imh     10201     363.9      346.      108.     7734.
      image.short.imh    262144     108.3     131.3      -1.    19936.
        new.short.imh    262144     108.3     131.3      -1.    19936.
```

17. Hopefully all went well to this point. Let's clean things up a bit.

```
cl> rew mta
cl> deall mta
cl> dir
check0001.imh       fitsout             image.sec.imh       new.short.imh
check0003.imh       image.dbl.imh       image.sect.imh
div.short.imh       image.real.imh      image.short.imh
```

Remember that if you want to delete any images you just use the task "imdelete". The task "delete" will delete your text files. If the wrong task is used to delete images a warning message is printed and no images are deleted.

If discrepancies occur during any of these steps, please look at the examples closely. It might be advisable to backtrack a few steps and verify things again. If the discrepancies are repeatable there could indeed be a problem. Please document the discrepancy and feel free to contact us if some advice or help is needed (iraf@noao.edu).