Name: _____

IRAF Exercise – Ast 399 – Spring 2003
Mon April 7, 2003

Today's class will be to get you familiar with IRAF. You should try to get through all these easy exercises by the end of the class period. I will help each group during the class. Iraf is set up to work on tinsley or cannon with the /a399 disk cross mounted. You can also run iraf on payne or leavitt, but as of now the disk is not cross mounted on those machines.

The data for this exercise is located on tinsley on /a399/data2.

1 Beginning Exercises

1. Set up your windows so that you have one window for iraf, a window for editing a file, and start ds9 (in any window type on a separate line ds9 9&). ds9 is the image viewer. Change directory to your iraf directory with

   cd usrname/iraf

   and start iraf (type cl).

2. In your other window, start up the vi editor to put the answers for the class exercise:

   vi exercise

   and type **i** to get into insert mode.

3. Change directories to the data directory. Type **ls** to get a listing of the files. This is a unix command which also works in iraf. More useful in iraf, however, is the "dir" command. Recall that in iraf, every command has a parameter file. Type

   lpar dir

   to see what the parameters are for this command. Create a long listing of the files in this directory. The syntax for commands follows the rules below:

   - You must give an answer for all parameters which are not in ( ). Others are optional.
   - To specify the answer to one of the parameters, put the answers on the command line in order after the command.
   - For optional parameters you can do these on the command line in any order, but you must give the parameter name. For example, if you need to specify the ncols value as 1, you would say **ncols=1** on the command line. You need not spell out the entire parameter word, just enough characters to be unique.
   - As a shortcut, you can let yes=+ and no=-

4. Recall that all files consist of a header with information about the file (this is in plain ascii text) and data (which is written in binary and is unreadable without computer decoding). The FITS header consists of keywords and values. Let's look at a header. To figure out functions which might be useful in iraf, there is a tool called **reference** (or refer) for short. Type

   refer name

where "name" is a keyword that you want to search on which might give you info about the package you are searching on. Find the package that will display the contents of an image header, and display the long listing of the header of image n2.10184.

5. You can see that there is a huge amount of information in the header. We would like to select out certain bits of information to make a neat summary table of the images. The function that does this is **hselect**. Make a table with the following information for all of the images, and put it in your *exercise* file for reference (using the mouse to cut and paste). Info: object name, type of observation, exposure time, filter, UT at start. Do this automatically, don't read the entries by hand and add them to the table! Note, in the hselect package, if you are retrieving more than one field, separate them by commas (no spaces). **$I** is the special name for the image name itself, which you always want to include.

6. Let's load an image into the image display. Use the reference command to figure out how to do this. Once you have found the function, list the parameters to see what the syntax is for displaying an image, and display image n2.10179 into image plane 1.

   (a) Just above the picture are 2 rows of buttons. The top row has the main commands, and the bottom row changes depending on what you select from the top row. Click on the **zoom** button, and then on the **out** button until the whole image is filling the screen.

   (b) Load image n2.10185 into image plane 2.

   (c) Zoom this image until it fills the display screen.

   (d) Select **Frame** in the top row of ds9 buttons, and then **blink** from the bottom row. Locate the comet. Move the mouse over the position of the comet and record it's position in your *exercise* file. You will want to turn off blinking (select **single**) and then put your mouse over the comet, centering it in the upper right zoom window, and then read off the x,y coordinates from ds9. You can use the right and left, up and down arrow keys on the keyboard to move the cursor with fine tuning.

   (e) How many counts are in the center of the comet in frame n2.10179?

7. Let's examine the image more closely. What function might tell you statistics about an image?

   (a) How big is this image (rows by columns)? (Hint use **imhead**).

   (b) If you want to act upon only a small section of an image, you can specify that section with the following syntax:
   image[x1:x2,y1:y2]

   (c) What is the maximum value found in image n2.10179 using the statistics function you discovered above?

   (d) Now find the maximum value in a small region 20 x 20 pixels in size centered on the comet nucleus in n2.10179. Write it in your *exercise* file. Is it different?

   (e) Display image n2.10179 again, but notice the information returned to you in the iraf window. What does it say? How does this compare to the max value you found for the comet using the two techniques above?

   (f) See if using lpar on the display function if you can figure out how to change the scaling of the image as displayed, so that when you put the mouse/cursor over the peak comet brightness it will display the correct value in ds9.

8. Let's look at two other ways to examine an image. The first one of these is called **imexamine**. The syntax is

   imexamine n2.10179

   Typing this command will bring up a new cursor in the ds9 window and you will be in *cursor mode*. Any commands you type now, should be typed while the cursor is over the ds9 window. If you need control of the cursor in the text window, it will automatically jump over there. Type a question mark in the ds9 window, and a list of all the cursor options will pop up in the text window – and cursor control will be in that window. To return the "attention" of iraf back to the ds9 window, keep hitting the space bar until the list is complete in the iraf window, then "q" to get out of the help mode. The round cursor will appear in the ds9 window again.

   (a) Make a surface plot centered on the comet

   (b) Make a radial profile plot, centered on the comet.

   (c) Sample some statistics in the vicinity of the comet.

   (d) Do a line plot through the comet

   (e) Do a row plot through the comet

   (f) Quit out of imexamine.

9. The second method is using **implot**

   (a) The function **implot** is a powerful plotting utility which is very useful for examining images, and is used heavily while reducing data. It is called with the following call:

   implot image

   (b) Run this function on n2.10179. Like the previous function there is a suite of cursor commands. Be careful not to confuse iraf with the position of the cursor. It will move the cursor control to the window that needs the control. In the tektronix window, type a question mark and see what the options are.

   (c) Make a column plot. The column can be roughly selected by moving your cursor right or left in the tektronix window to correspond to the column you wand to plot.

   (d) For really precise control of which column to plot, use a colon command (see the help list generated with the ?). Plot 3 columns centered on the comet nucleus.

   (e) Print the cursor position at the peak point of the comet - what is the value?

   (f) Average all 2100 lines of the ccd together in a line plot

   (g) Display the y axis value from 0 to 3000. Display the x axis value from 2000 to 2100.

   (h) The low values at the right are the bias columns. Change the y and x display ranges to get the best estimate of the bias level. What is the bias level, and between which columns is it best measured?

10. Let's learn how to combine images – this is necessary for image processing. Use the **refer** command to figure out what the best function would be for combining images. Combine all the Tempel 1 images into a single image using a median combine.

2 Intermediate Exercise

1. Make a bad pixel map for this detector. This is a file which lists the positions of all the bad (dead, non-flattening) pixels in the detector. Discuss options you may have for making such a map, and what the pluses and minuses are for each technique. (i.e. there is more than one way to do this).

   (a) When it comes to actually noting down the values of the bad pixels, see if you can figure out a way to do this using the cursor rather than writing them down, or typing them.

3 Advanced Excercise

1. Flatten the data. Much of your task has been made easier because the flats are already prepared.

   (a) Using the following ccd.dat file, check the image type on all the files to make sure there are no errors.

   exptime exptime
   darktime darktime
   subset filters
   biassec biassec [407:414,58:557]
   trimsec datasec [23:402,58:557]
   imagetyp imagetyp

   'OBJECT' object
   'FOCUS' object
   'MULTIPLE EXP' object
   'COMPARISON' object
   'BIAS' zero
   'DOME FLAT' flat
   'FLAT' flat
   'SKY FLAT' flat

   fixpix bp-flag 0
   overscan bt-flag 0
   zerocor bi-flag 0
   darkcor dk-flag 0
   flatcor ff-flag 0
   fringcor fr-flag 0

   (b) Determine the bias and trim sections. CAUTION - because these flats have been processed, the bias and trim must be identical to the processed flats. After getting the bias and trim sections, edit the ccd.dat file to match, and edit the parameter files for ccdproc to reflect the correct values.

   (c) Run ccdproc to fix the bad pixels in all images

   (d) Determine the overscan fit, testing manually for the best result, then do the overscan correction and trim the images.

(e) Do the zero corrections

(f) Flatten the images - and test to see how good the flattening is.

(g) Clean the cosmic rays from the images using the following recipe:

```
Load noao.nproto

Edit gain and readnoise to be sure correct (1.74,6) (findthresh)

Create a procedure of the form:

        imstat image fields="mode" | findthresh image=image ver-


        join lsobject lsobject > proccr1a
        cl < proccr1a > outcr1a &

Odd problem - it was crashing.  It didn't like the pipe output -
had to
do this into 2 pieces

        cl < proccr1a1 > outcr1a1 &  Then edit outcr1a1 for findthresh
        cl < proccr1a2 > outcr1a

        lintran outcr1a xs=4 > outcr1b
        edit to remove the second column in outcr1b

Create a procedure of the form:

        cosmicrays image image thresh=(from outcr1b)

        join lsobject lsobject > temp
        join temp outcr1b > proccr1b (edit)
        cl < proccr1b &

Testing                         Threshold Levels      Cosmic Rays Removed
Frame            #       Z=5   Z=4.5   Z=4    Z=3   Z=5   Z=4.5   Z=4
n1.10052 900 R 42.701  213.50 192.15 170.80 128.10  3357  3596  4311  17
hsel test* $I,CRCOR yes

Use threshold = 4
Don't let any of the thresholds go below 60.
```