Assignment #1 Notes

Getting Started in Lab

(1) Use `Fetch` to download the following three subdirectories from the course directory on the CS file server to your Desktop:

```
/home/cs332/download/edges
/home/cs332/download/assign1
/home/cs332/download/assign1images
```

(2) In MATLAB, set the Current Directory to the `assign1` folder

(3) In MATLAB, add the `edges` and `assign1images` folders to the MATLAB *search path:*

(a) from the **File** menu, select **Set Path…**
(b) in the dialog box that appears, click on the **Add Folder…** button
(c) in the browser window that appears, navigate to the place where you see the `edges` folder listed, select it with a click, and then click on the **Open** button
(d) repeat steps (b) and (c) for the `assign1images` folder
(e) close the **Set Path** window - you will be asked whether you want to save the changes for future MATLAB sessions – click on **No** (MATLAB won't actually let you save)

Finishing Up

At the end of lab, upload your `assign1` folder to your personal directory on the CS file server. If you are uploading a folder for this class for the first time, it would be good to create a `cs332` subdirectory for storing all of your work for this class. To do this, create a new folder on your Desktop named `cs332` and drag your `assign1` folder into the `cs332` folder. Then connect to your personal directory on the CS file server using `Fetch`, and upload the entire `cs332` folder to this directory. In the future, you can upload all personal code folders for this class into your `cs332` subdirectory.

Continuing your work on this assignment at a later time

In the future, you should again download the `edges` and `assign1images` folders from the course directory as described above, but download the `assign1` folder from your personal directory. Also repeat steps (2) and (3) above. If you make changes to the code files in your `assign1` folder, be sure to upload the modified files onto your personal directory.

Problem 1: Analyzing intensity changes in a two-dimensional image

```
edit yachtScript.m          % open yachtScript.m in the editor
yachtScript;                % run the script file
```

Drag apart the Image Tool windows to view them separately, and use the Pixel Region tool to view the contents in more detail. Click on the zoom-in or zoom-out icons in the Image Tool window to turn zooming on or off.

```
displayImage(zc4,8);        % display the slopes of the zero-crossings
displayImage(zc8,16);
```

A few MATLAB programming tips:

Terminating a program:
When a MATLAB program is running, the word "Busy" appears in the lower left corner of the main MATLAB window. Execution can *usually* be terminated by typing **control-C**.

Setting the range of values for `imshow` and `imtool`
When given a 2D matrix of `uint8` values as input, `imshow` and `imtool` automatically assume that the values range from 0 to 255, and display 0 as black and 255 as white, with intermediate values displayed as shades of gray. A different range of values can be specified when calling these functions:

```
>> imshow(image, [50 100])
>> imtool(image, [50 100])
```

In this case, values in the input image that fall below 50 are displayed as black, values above 100 are white, and values between 50 and 100 are displayed as shades of gray from black to white.

When given a 2D matrix of `double` values as input, `imshow` and `imtool` automatically assume that the values range from 0.0 to 1.0, and display 0.0 as black and 1.0 as white, with values between 0.0 and 1.0 shown as shades of gray. Again, a different range of values can be specified:

```
>> imshow(image, [-10.0 10.0])
>> imtool(image, [100.0 500.0])
```

`sum, prod, min` and `max` of a 2D matrix:
When given a 2D matrix as input, these functions return a 1D matrix of sum, product, minimum or maximum values obtained for each column of the 2D matrix. To obtain a single result for the full 2D matrix of values, these functions need to be called twice:

```
>> vals = [1 6 2; 3 0 -2]
vals =
      1    6    2
      3    0   -2
>> sum(vals)
ans =
      4    6    0
>> sum(sum(vals))
ans =
      10
>> min(vals)
      1    0   -2
>> min(min(vals))
ans =
      -2
```