CS30 Spring 2014
Lab 5

Use the command `diary` to record your answers and submit them. Submit code for the scripts and functions you write. Submit any figures.

For the problems below, define

```
rowArray = [ 1, 2, 0, 3, -1, 0, 0 ];
colArray = rowArray';
twoDimArray = [ 0, -1, 2; 4, 0, 1 ];
```

1. (30 points) 2D arrays.

   (a) Flatten `twoDimArray` into a column array `flatArray`.

   (b) Copy `twoDimArray` into `twoDimArray2`. Replace all elements of `twoDimArray2` with the value -1.

   (c) Copy `twoDimArray` into `twoDimArray2`. Delete the third column of `twoDimArray2`.

   (d) Copy `twoDimArray` into `twoDimArray2`. Delete the second row of `twoDimArray2`.

   (e) Copy `twoDimArray` into `twoDimArray2`. Delete the element of `twoDimArray2` at linear indices 3 and 4. What size is the resulting array?

   (f) Get all the elements of `twoDimArray` that are greater than 0 using a relational expression and logical indexing.

   (g) Get all the elements of `twoDimArray` that are greater than 0 and less than 3.

2. (40 points) Matlab's `find` function takes an array and returns the linear indices of the non-zero elements. If the input array is a 1D row (column) array, the result is a row (column) array. If the input is a 2D array, the result is a 1D column array.

   (a) Use the `find` function to find the non-zero elements of `rowArray`, `colArray`, and `twoDimArray`.

   (b) `find` can be used to find element indices satisfying other relations. For example, to find elements of `rowArray` larger than 2, run

   ```
   find(rowArray > 2)
   ```

   The relational expression `rowArray > 2` creates a logical array the size of `rowArray` with 1 in the element positions satisfying the relation and 0 in the other element positions. This logical array is then passed to `find` which returns the linear indices of the non-zero elements. Use `find` to get the indices of all elements of `rowArray` that are greater than 0.

(c) Use `find` to get the indices of all elements of `twoDimArray` that are greater than 0 but less than 3.

(d) Use `find` to get the indices of all elements of `colArray` that even or negative.

(e) Write your own function `MyFind` which takes as input a single 1D or 2D array and finds the linear indices of the non-zero elements. If the input is a 1D row (column) array, the output should be a 1D row (column) array. If the input is a 2D array, the output should be a 1D column array. Do not use Matlab's `find` function in your implementation.

(f) Run your function on `rowArray`, `colArray`, and `twoDimArray` as you did with Matlab's builtin `find` function and confirm that you have the same results.

3. (30 points) 2D Image array. Write a script called `MirrorGrayImage.m` to create a mirrored grayscale image. Your scripts should implement the following steps.

(a) Read in the image `UCRColor.png` into a variable `ucRGBImage` by using the `imread` function. Note that this generates a three-dimensional array, where the first two dimensions are the row and column of the pixel respectively, and the third dimension contains the RGB values of the pixel.

(b) Set Matlab's image colormap to 'Gray' using the `colormap` command.

(c) Average the RGB values in the third dimensions to get a single grayscale value. Use the matlab function `sum` to compute the average. Name the resulting array `ucGrayImage`. Display the image using the `image` command. It should match the given image `UCRGray.fig`.

(d) Create an image which mirrors `ucGrayImage` left-to-right and top-to-bottom as shown. Name it `mirrorGrayImage` and display it using the `image` command. It should match the given image `UCRGrayMirrored.fig`.

(e) Display the transpose of the image.