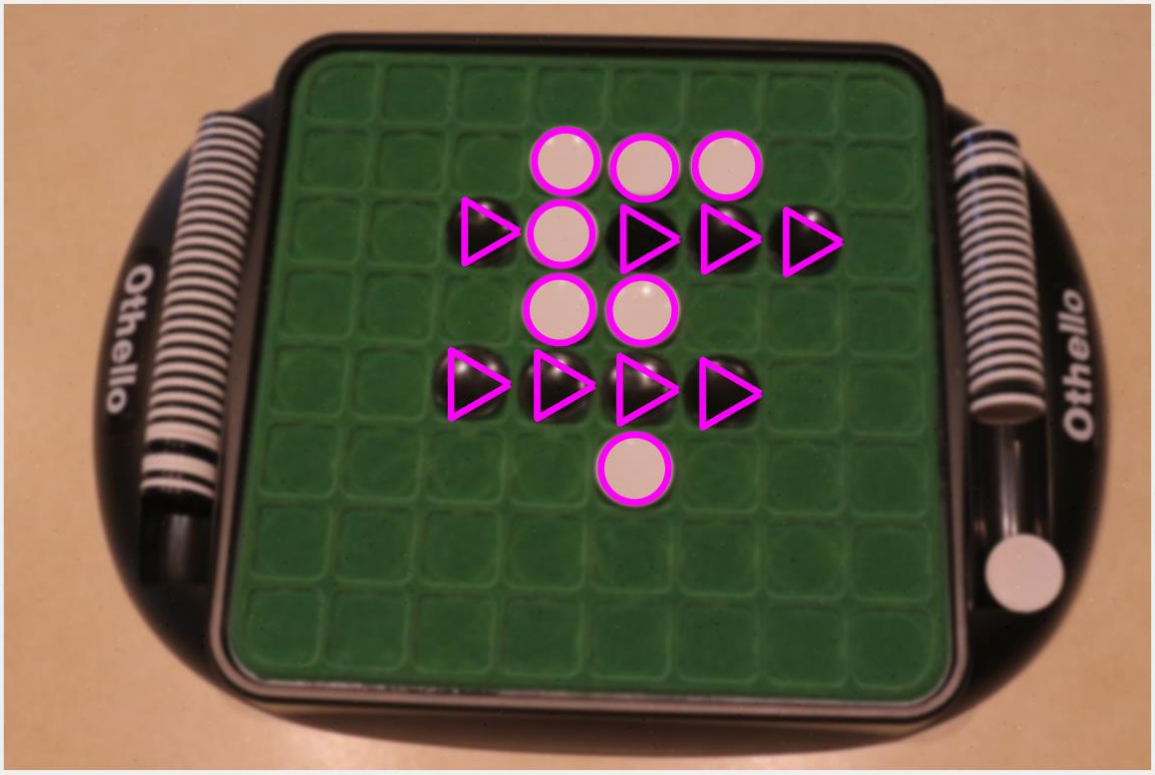
1. MATLAB given code answers
    a. Part 1
    magic.' Gives the transpose of the matrix m3
    size(mstacked) returns 3,6 implying 3 rows and 6 columns
    b. Part 2
    Imshow shows the image at 100% magnification and doesn't work accurately with values however, imagesc puts a default axis and works seamlessly with values. Colormap sets the color structure of the current figure to the value that has been passed to it, like 'hot', 'gray', etc. It returns a 3-column matrix of RGB triplets defining the colormap for the current figure.
    c. Part 3
    M3^2 gives the product rule of the matrix whereas m3.^2 gives the square value of each number in the matrix and maintains that matrix.
    Shorthand for m3 * m3 is m3^2
    Mstacked(3,2) = 9
    Size(Mstacked, 1) = 3
    Size(Mstacked, 2) = 6
    Length(mstacked) = 6
    The status by running the following command:
    Disp('Try the following command and report what results')
    Gives out the following output: 23, 31, 43, >>
    Where the cursor is next to the output as there is no new line character being printed in the above output.
    The loop is iterated 6 times. The matrix gets printed out in the transposed form. The reason for this is that MATLAB is column major, which implies that it will address an entire column and its rows and then move to the next column.
    d. Part 4
    Im_smaller is all the green bit value of the of the original image, thus appearing gray, 0.5 times the original image.
    The height of each shade indicates the value of the pixel. The value would be in the range of 0 to 255.
    View(150,50) sets the angle of the view from which an observer see the current 3-D. 150 implies the horizontal rotation and 50 gives the vertical elevation.
2. Things I learnt from the examples:
3. For the Othello board, the following were the results:
    a. The number of white pieces were 8
    b. The number of black pieces were 7

    In order to address this experiment to identify the number of individual pieces, I first tried to isolate the channels from the original image and compared it with the gray scale image. After doing this, I realized that the red channel resembled the closest, so I graphed a histogram for it to identify the places where it had spikes and low values. The spikes would indicate the white circles and the low values would have indicated the dark circles.

After that I identified the specific values that would act as thresholds for the respective colors, however, while trying to use the 'imfindcircles()' code for the individual set of shapes, I wasn't able to get the total amount of circles for the specific colors.

I changed my approach and decided to work on the gray scale image instead. I used the concept of Opening on the gray scale image of the Othello board. I first made sure that the board had been adjusted with a sequence of 0s and 1s by getting a threshold to dictate that working. From the Opening, I still wasn't able to get the circles as I wanted as a few were still getting left out for the whites.

I changed my approach and decided to graph a histogram on the gray scale. I got new thresholds for white and black circles. I then entered each of the sets to 'imfindcircles()' to indentify the respective colored circles and I was able to get all the respective circles on the board. The image for that is shown below where the white circles are circled with circles and the black circles are marked with triangles on them.




Overall, I am not happy with the way this experiment turned out for me as I wasn't able to use the actual concepts that were taught in the class. Towards the end, this experiment appeared to be more related to a trial and error approach and concluded with a brute-force method. This implies that I still need to look more into the concepts of Opening and Closing and how exactly they should be implemented correctly.