**Question#1**

**Part 1: Code**

**ConnectedNeighbors Function:**

The starting position(pixel) is given. Check neighboring pixel values against threshold value. There are 4 neighbors two horizontal neighbors and two vertical neighbors. If the difference is in threshold then and are not visited then they are saved in array.

The code is also attached.

**ConnectedSet Function:**

In this function, I have called the connectedNeighbors Function and get the neighbors for the pixel which I have passed in it. When I visit any pixel, I change the corresponding position value to 1 which keep track of visited items. Assign different labels to different segments.

**Part 2: Output**

This is the output for pixel = (45,67) using Threshold = 2.

The pixel which are connected with threshold=2 are set to 0(Black) and others set to 255 (white).

C:\Users\Saddam\AppData\Local\Microsoft\Windows\INetCache\Content.Word\img22gd2.tif

Original Image Output image

**Part 3:**

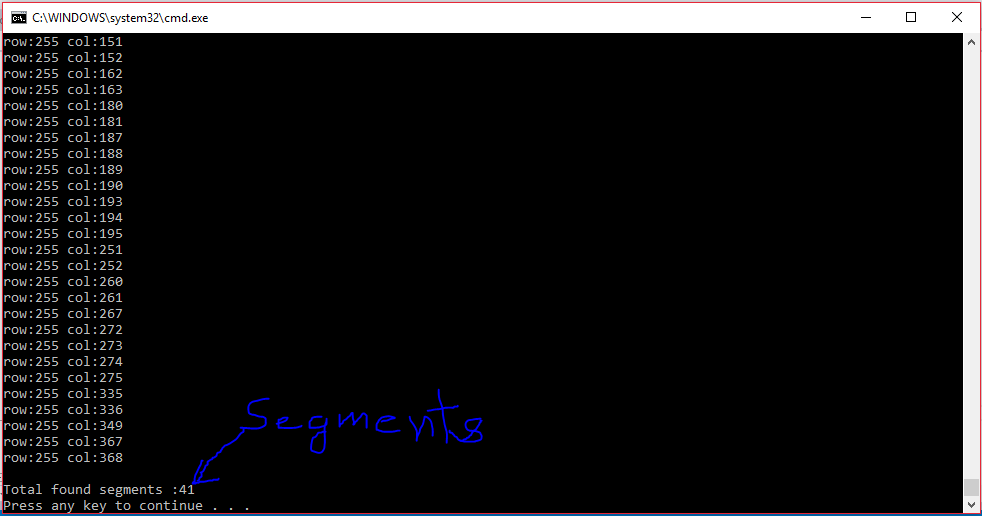
I have generated a segmentation of the image as explained in Section 2 of CCSeg.pdf. I have used raster way to find all the segments in the image. I have found 41 segments.

**Input**

C:\Users\Saddam\AppData\Local\Microsoft\Windows\INetCache\Content.Word\img22gd2.tif

**Output Intermediate Result**

Starting label Values from **1, It is Visual Studio’s Output Segments**





**Output of Segmentation with different colors:**

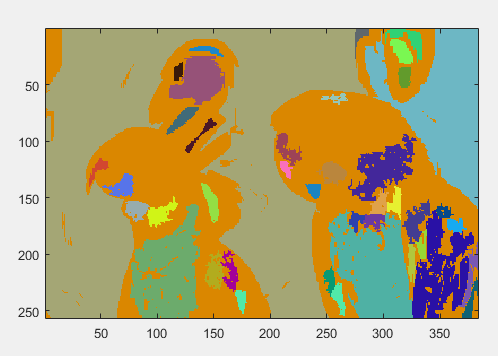
x=imread('color.tif');

N=max(x(:));

image(x);

colormap(rand(N,3));

axis('image');



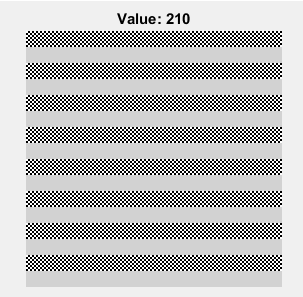
**Question#2**

**Part 1:**

Image of matching gray level.

First I set the black level of the screen (monitor).

I have started from 0 level to 255 and found minimum difference between the checker board line and line having gray level 127.

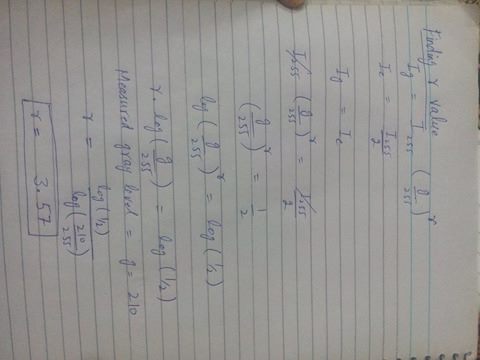


**Part 2:**

**Derivation:**

**The level I found 210**

**I got gamma = 3.57**



**Part 3:**

img = imread ('C:\Users\Saddam\Desktop\Digital Image processing\Project 2\Data\linear\linear.tif');

imshow(img);

%This is my gamma value

gamma = 3.57;

corLinearImg = 255\*(double(img)/255).^(double(1/3.57));

figure(2);

image(uint8(corLinearImg)+1);

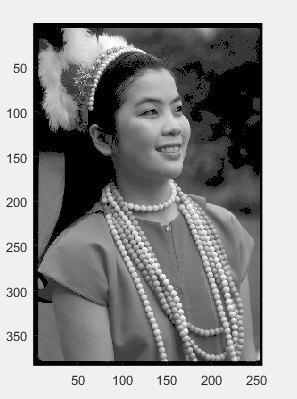
set(gca, 'Box', 'off');

axis('image');

graymap = [0:255;0:255;0:255]'/255;

colormap(graymap);

**Input Image Output Gamma Corrected Image**

C:\Users\Saddam\AppData\Local\Microsoft\Windows\INetCache\Content.Word\linear.tif

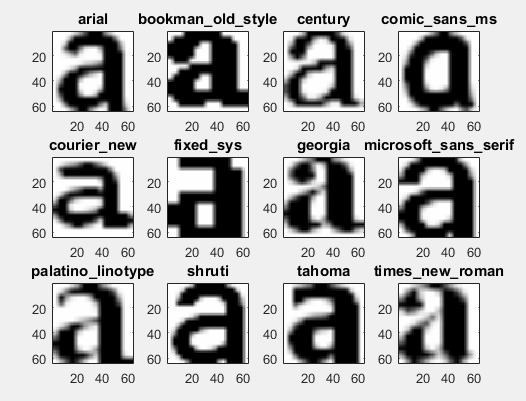
**Question#3**

**Part 1:**

**Vectorized Image:**

**This is the vectorized data as explained in section 4.1.**

**All the images are in column form saved.**



**Part 2:**

**Find mean**

%calculate the mean of each coloumn

mean\_X = mean(X,2);

**Find the centered image**

center\_image = (1/sqrt(n-1))\*(bsxfun(@minus,X,mean\_X));

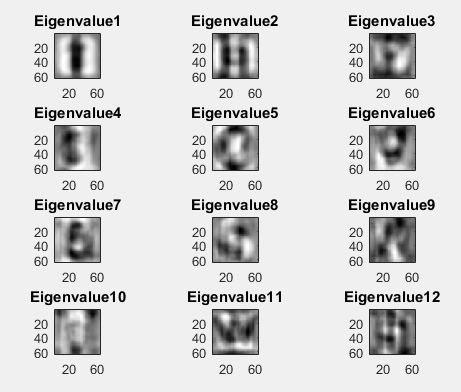
**Find the Singular Value Decomposition**

%Singular Value Decomposition

[U,S,V] = svd (center\_image, 0);

**Then print the eigen vectors**

**12 Eigen vectors**



**Resynthesized versions of the original image**

m = 1;

for k = [1 5 10 15 20 30]

Xtmp = U(:,1:k)\*Y(1:k,:);

Xest = bsxfun(@plus,Xtmp,mean\_X);

img = reshape(Xest(:,1),64,64);

figure(4); subplot(3,2,m); image(img);

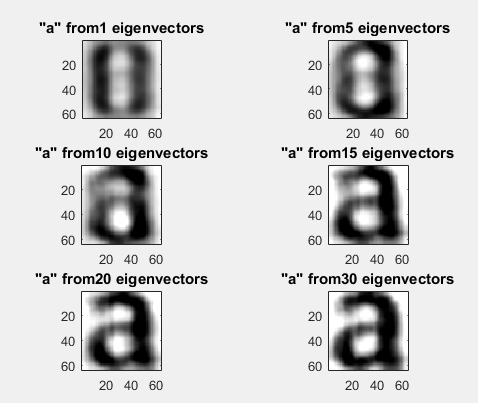
title(strcat('"a" from ',num2str(k),' eigenvectors'));

axis('image'); colormap(gray(256));

m = m+1;

end

**Output:**



**Graph:**

**Projection coefficients vs. eigenvector numbers**

Y = U'\*(bsxfun(@minus,X,mean\_X));

figure(2);

for k = 1:4

plot(1:10,Y(1:10,k));

hold on;

end

legend('a','b','c','d');

xlabel('Eigenvector number');

ylabel('Projection value');

**Output:**

