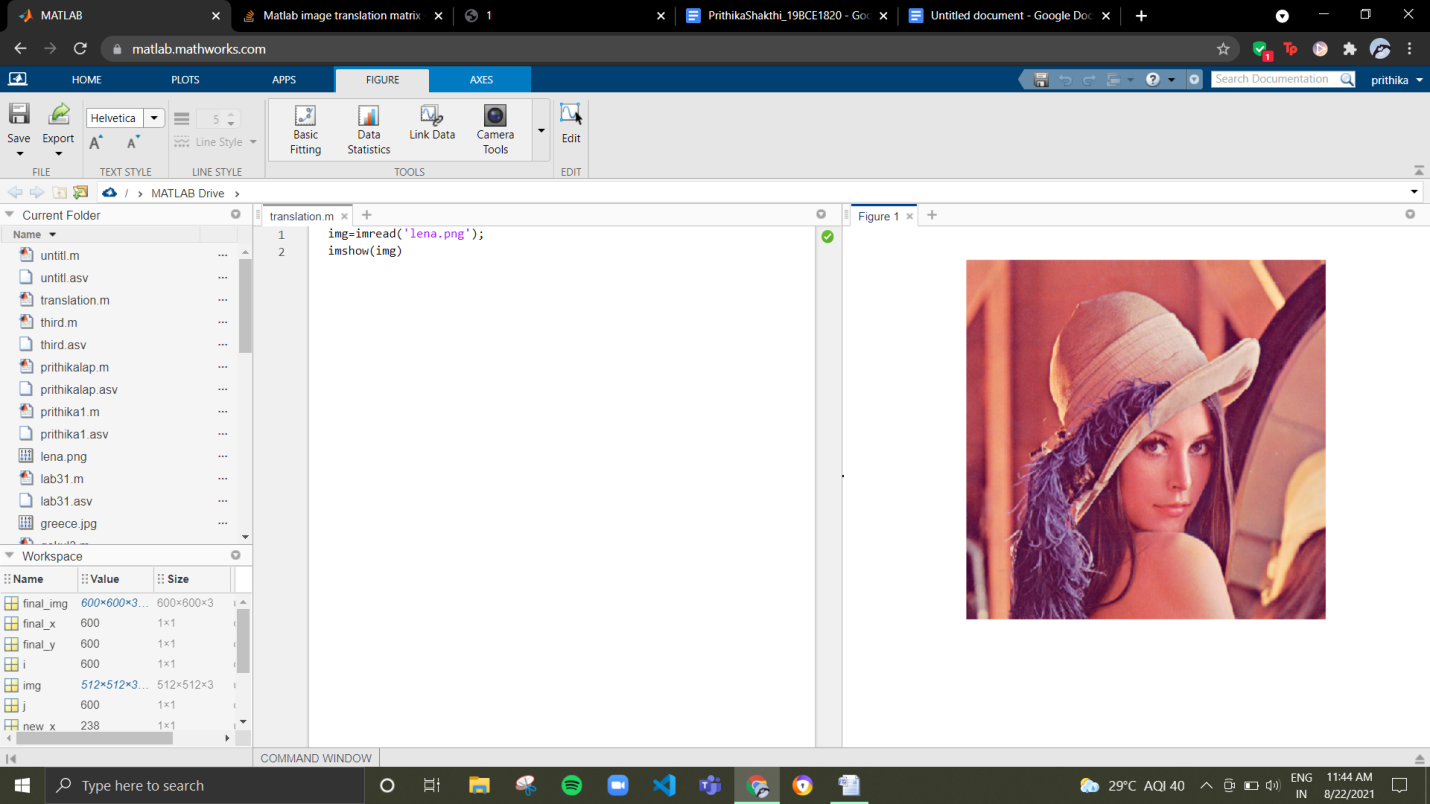
**Image Used: Lenna Image**

****

**(i) Translation of an image**

**Aim:** To perform translation to an image

**Algorithm:**

Step1: Read the image

Step2: Initialize the number of row and column values to move

Step4: Create the filter

Step5: Obtain the translated image.

**Code:**

**img=imread('lena.png');**

**x = 20;**

**y = 20;**

**q = [size(I,1); size(I,2); 1];**

**dia = [0 1 x; 1 1 y; 1 0 1];**

**q2 = mtimes(dia,q);**

**imgnew = uint8( zeros(q2(1)-1, q2(1)-1, size(I,3)));**

**imgnew(y:end, x:end, :) = I;**

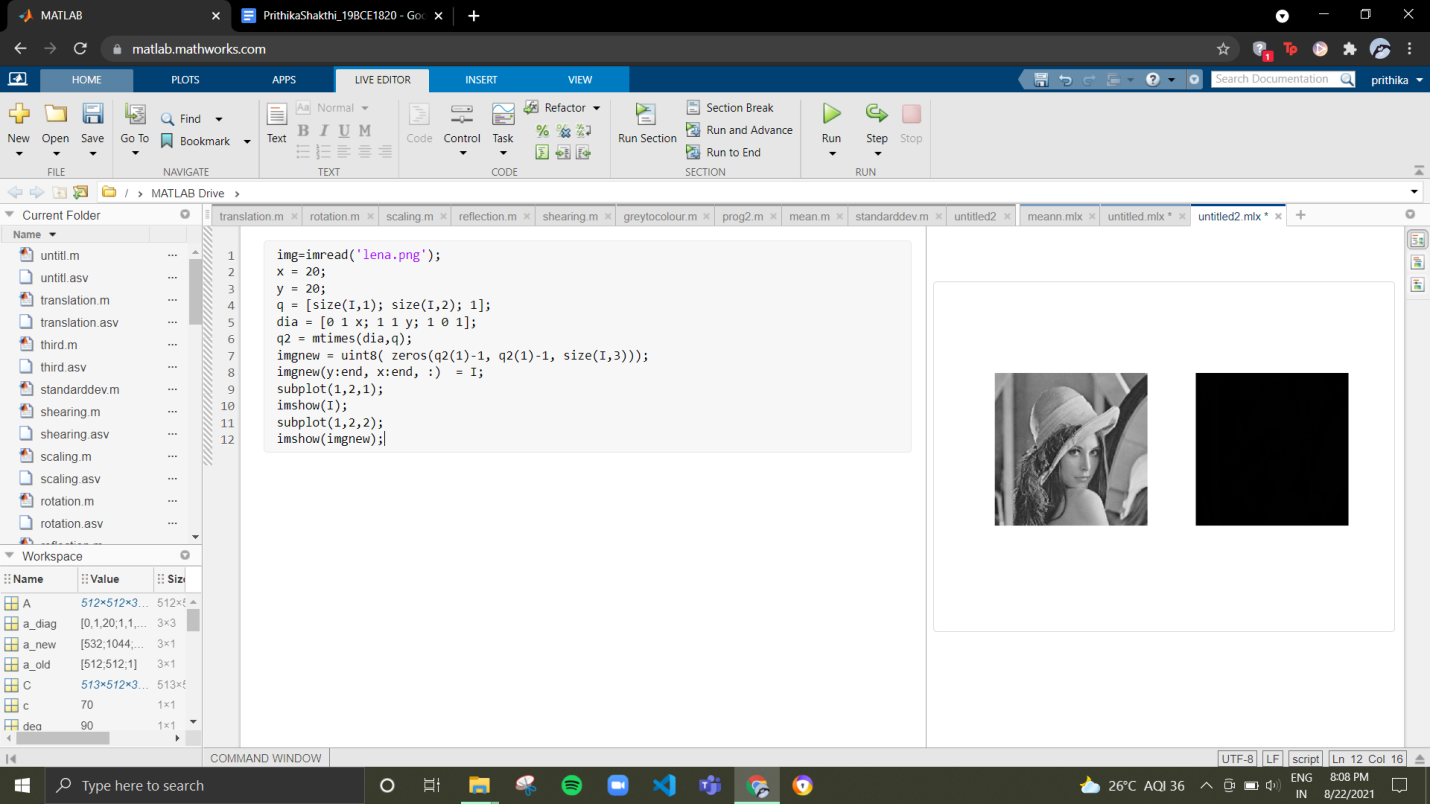
**subplot(1,2,1);**

**imshow(I);**

**subplot(1,2,2);**

**imshow(imgnew);**

**Screenshot:**

****

**(ii) Rotation of an image**

**Algorithm:**

Step1: Clear all the text from the command window

Step2: Clear all from the workspace.

Step3: Read the original image and save it as “og”

Step4: The original image is rotated by an angle theta taking midpoint as reference point

Step5: The pixel coordinate is converted into polar coordinates.

Step6: Convert angle in radians and radius

Step7:Then convert angle theta to degrees.

Step8: Rotation angle is added to the theta

Step9: The angle is converted back to radians

Step10: Angle is now converted to obtain the polar coordinates of the resultant rotation matrix

Step11: The matrices are then checked to make sure than all their elements lie within the range of the image size

**Code:**

img=imread('lena.png');

a1=zeros([size(img,1)\*size(img,2) 1]);

a2=zeros([size(img,2)\*size(img,1) 1]);

deg=90;

C=uint8(zeros([size(img,1) size(img,2) 3 ]));

m=1;

midx=ceil((size(img2,1)+1)/2);

midy=ceil((size(img2,2)+1)/2);

for i=1:size(A,1)

i1=i-midx;

for j=1:size(A,2)

[t,r]=cart2pol(i1,j-midy);

t1=radtodeg(t)+deg;

t=degtorad(t1);

[x,y]=pol2cart(t,r);

a1(m)=round(x+midx);

a2(m)=round(y+midy);

m=m+1;

end

end

a1(find(x1 < 1))=1;

a2(find(x2 < 1))=1;

n=1;

for i=1:size(A,1)

for j=1:size(A,2)

img2(x1(n),x2(n),:)=A(i,j,:);

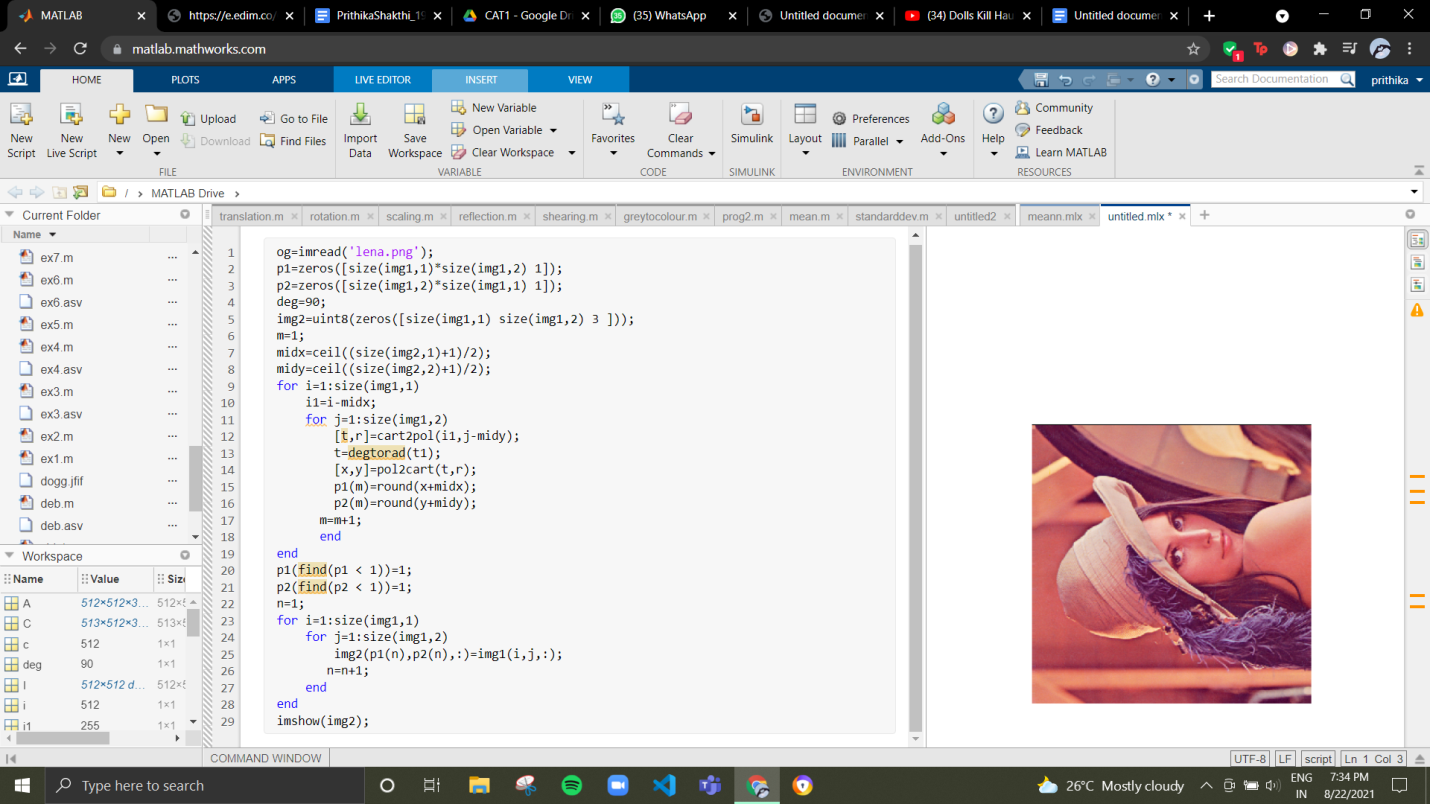
n=n+1;

end

end

imshow(img2);

**Screenshot:**

****

**(iii) Scaling of an image**

**Code:**

clc;

clear;

img=imread('lena.png');

q=size(img,1);

q2=size(img,2);

a=1;

b=1;

for i=1:q

for j=1:q2

b2(a,b)=img(i,j);

x2=x2+1;

b2(a,b)=img(i,j);

b=b+1;

enda

a=a+1;

b=1;

for j=1:q2

b(a,b)=img(i,j);

b=b+1;

b(a,b)=img(i,j);

b=b+1;

end

a=a+1;

b=1;

end

a=1;

b=1;

for i=1:2:q

for j=1:2:a2

d(a,b)=img(i,j);

b=b+1;

end

a=a+1;

b=1;

end

c=uint8(b);

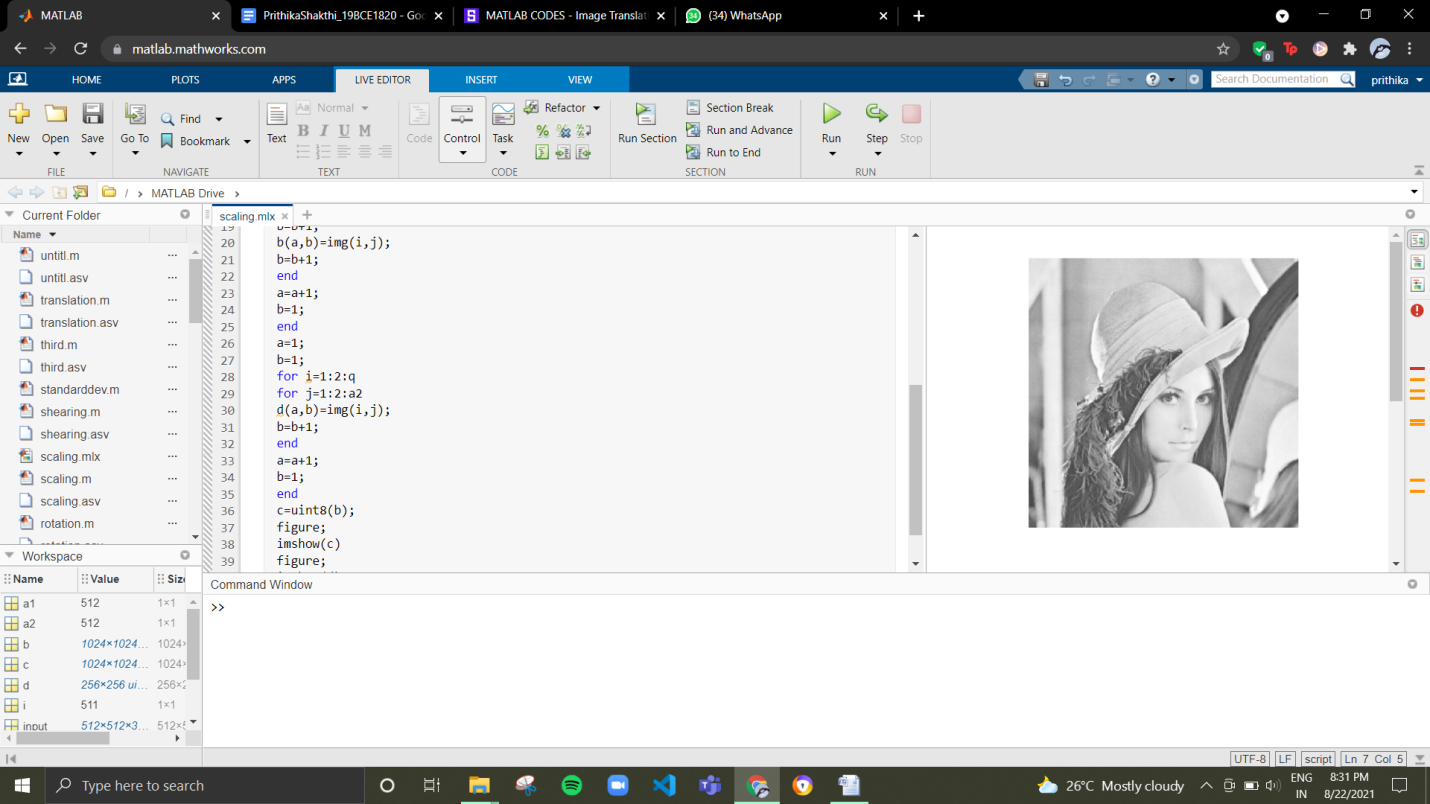
figure;

imshow(c)

figure;

imshow(d);

**Screenshot:**

****

**(iv)Reflection of an image**

**Code:**

clc

clear all

img=imread('lena.png');

p=size(img,1);

q=size(img,2);

for j=1:q

img2(:,q-j+1,:)=img(:,j,:);

enda

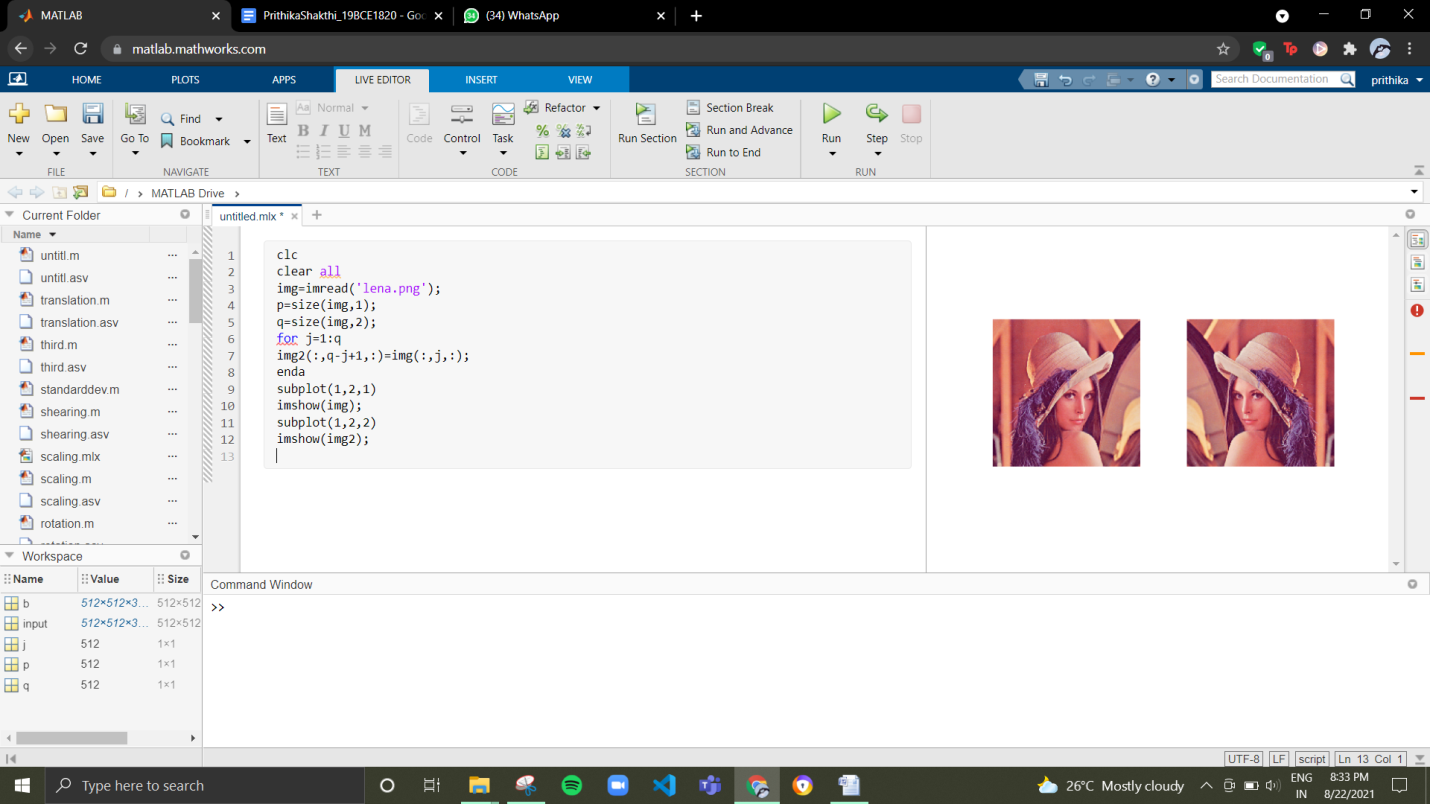
subplot(1,2,1)

imshow(img);

subplot(1,2,2)

imshow(img2);

**Screenshot:**

****

**(v) Shearing of an image**

**Code:**

clc;

clear all;

img=imread('lena.png');

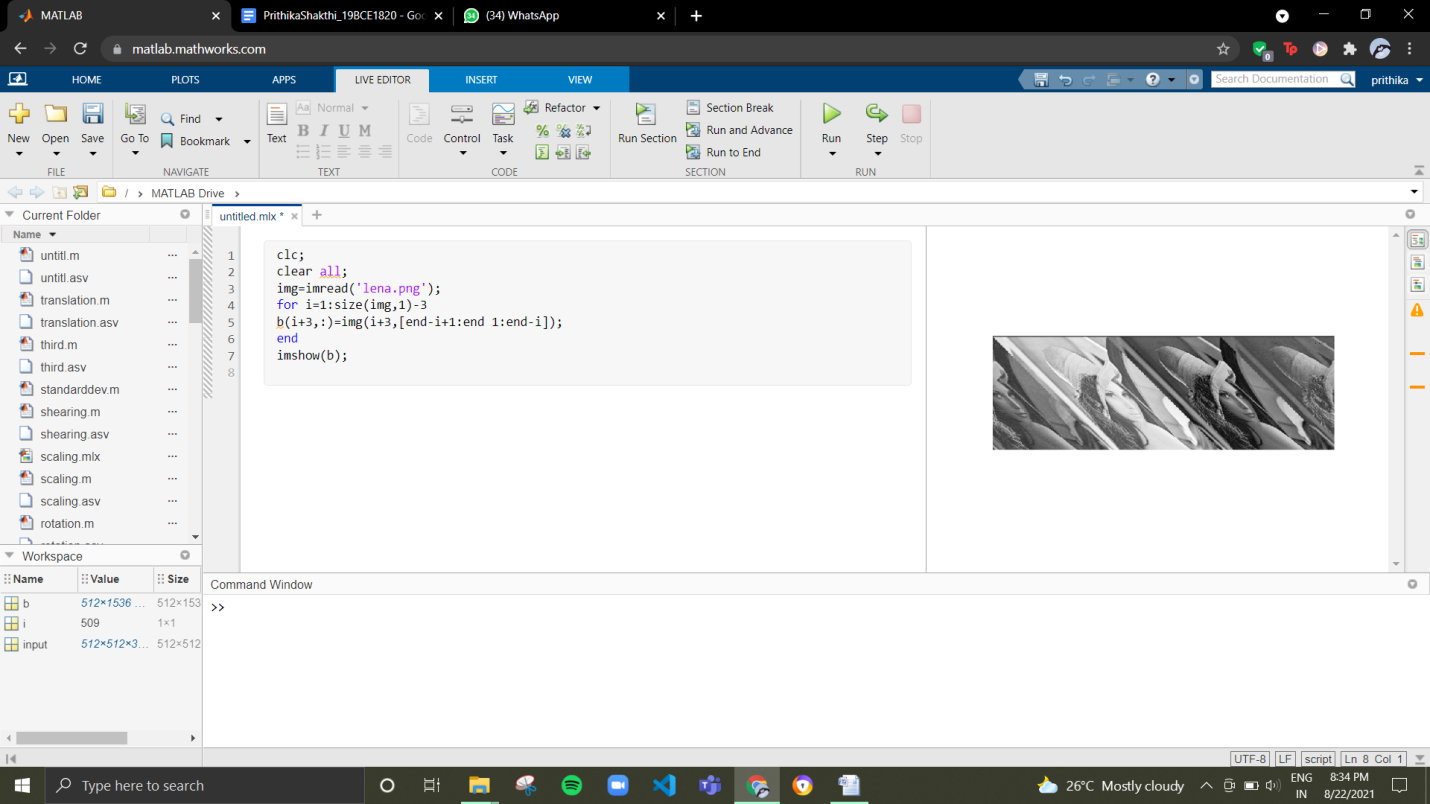
for i=1:size(img,1)-3

b(i+3,:)=img(i+3,[end-i+1:end 1:end-i]);

end

imshow(b);

**Screenshot:**

****

**(vi)Eigen value of an image**

**Algorithm:**

**Step1: Clear the command window**

**Step2: Read the image and store it in img**

**Step3: Get the size of the image**

**Step4: Compute the square matrix and assign the values**

**Step5: Display the eigen values**

**Code:**

clc;

clear all;

img=imread('lena.png');

b=size(img);

a=max(b);

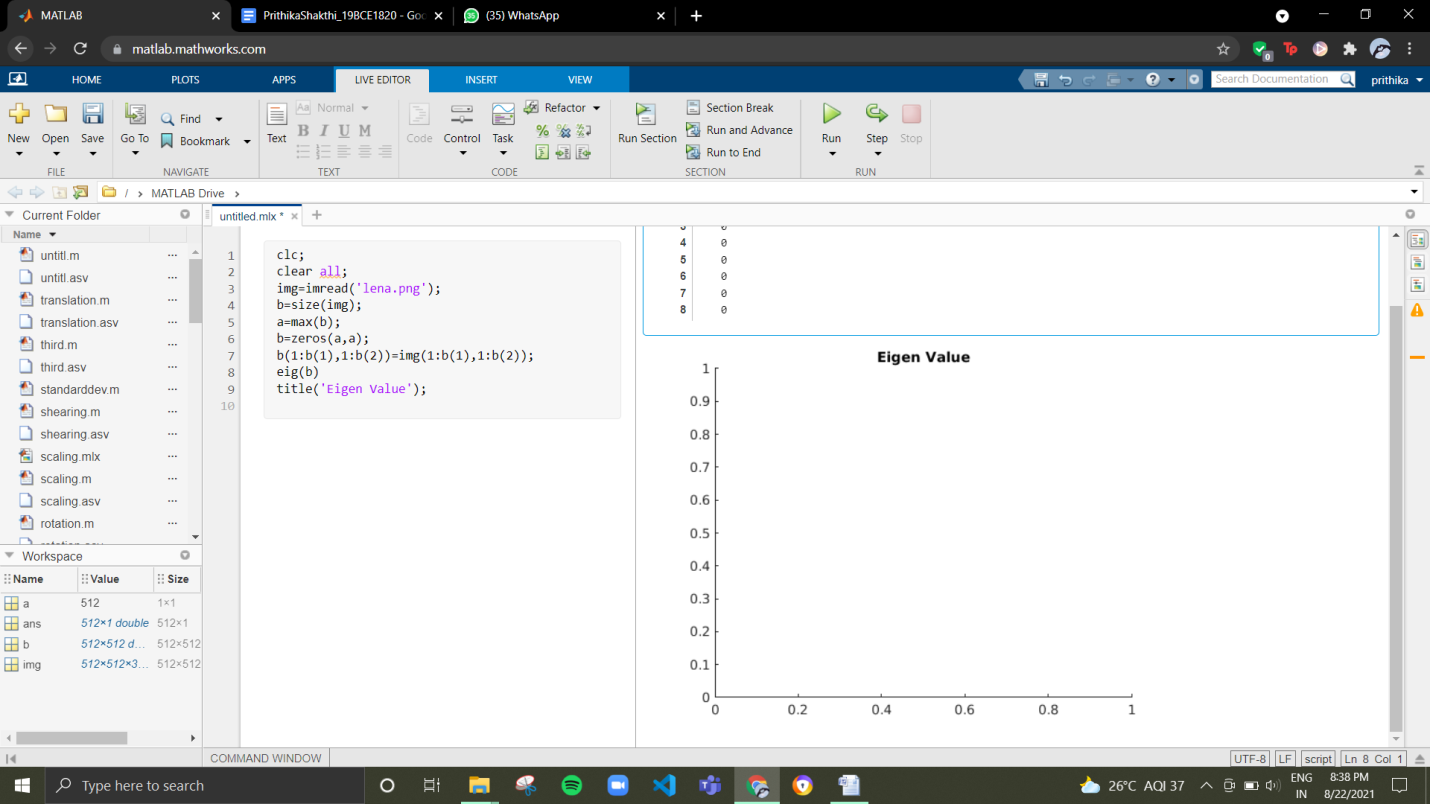
b=zeros(a,a);

b(1:b(1),1:b(2))=img(1:b(1),1:b(2));

eig(b)

title('Eigen Value');

**Screenshot:**

****

**(vii) Square of an image**

**Code:**

clc;

clear all;

img=imread('lena.png');

img=double(img);

a=size(img);

x=uint8(zeros(size(img)));

for i = 1 : size(img, 1)

for j = 1 : size(img, 2)

a(i, j) = img(i,j).^2;

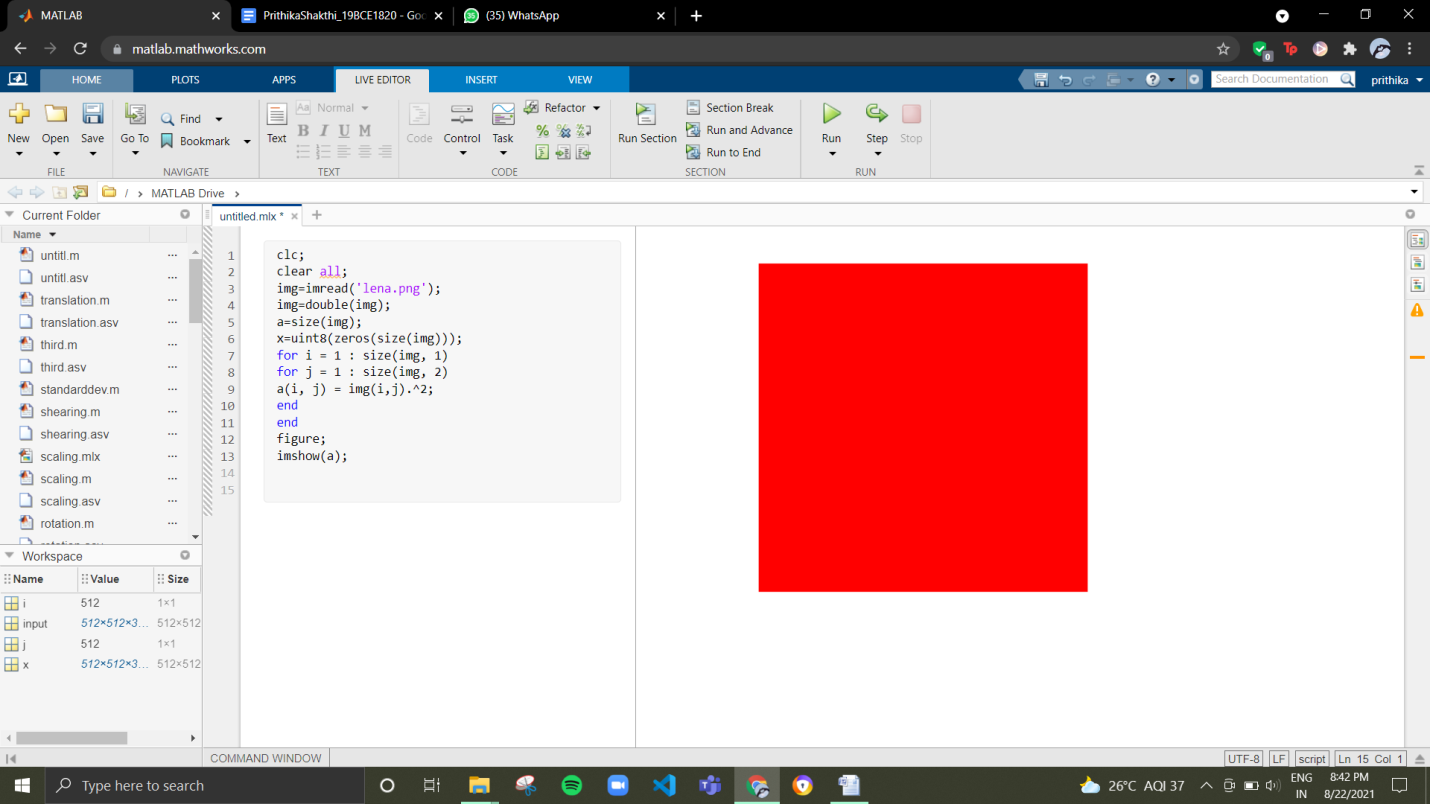
end

end

figure;

imshow(a);

**Screenshot:**

****

**(viii) Square root of an image**

**Code:**

clc;

clear all;

img=imread('lena.png');

img=double(img);

x=size(img);

z=double(zeros(size(img)));

for i = 1 : size(img, 1)

for j = 1 : size(img, 2)

z(i, j) = sqrt(img(i,j));

end

end

figure;

imshow(z);

**Screenshot:**

****

**(ix) Convert Greyscale into colour**

**Code:**

og = rgb2gray(imread('lenag.png'));

og2 = zeros([size(img,1) size(img,2) 3]);

map = colormap(jet(256));

R = map(:,1);

G = map(:,2);

B = map(:,3);

og2(:,:,1) = R(og);

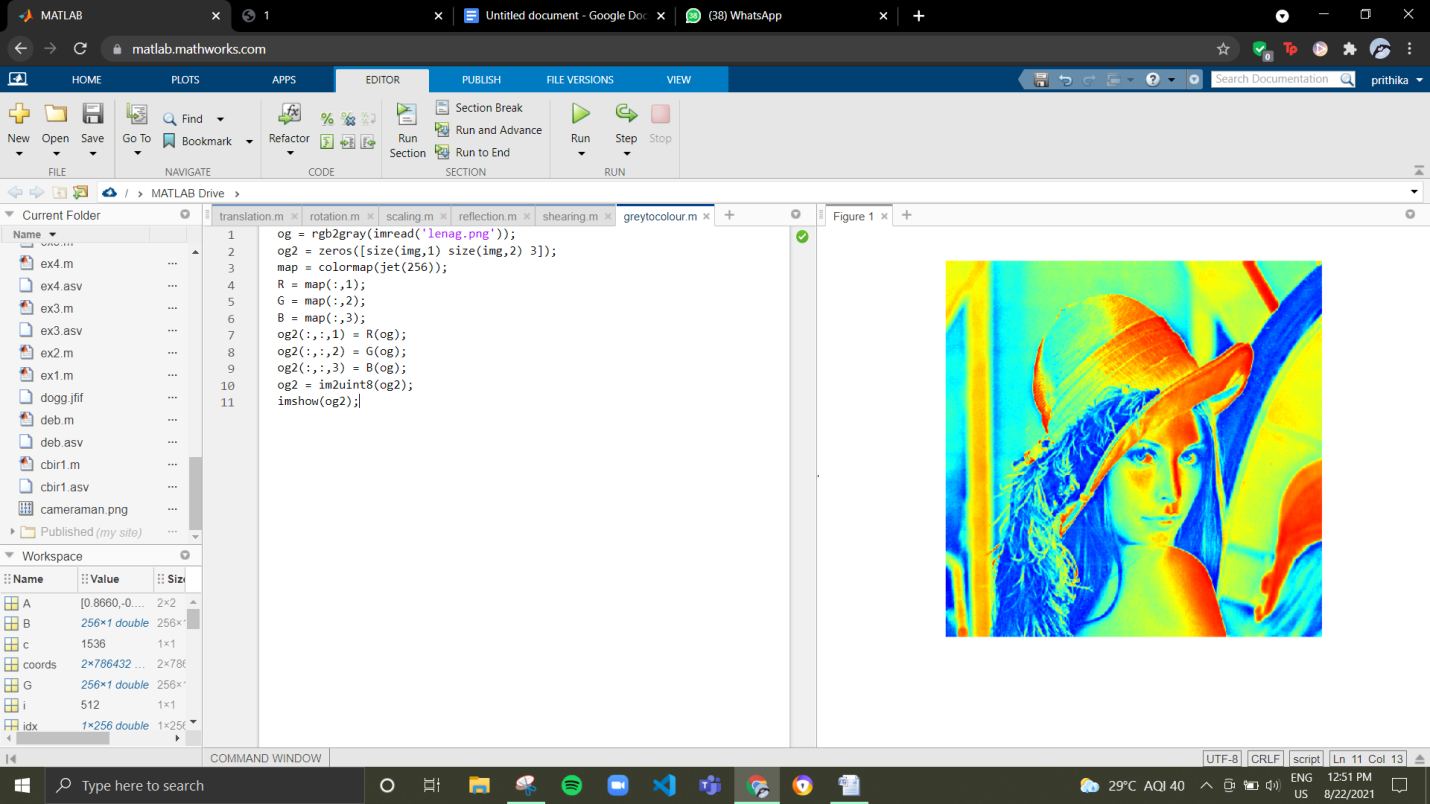
og2(:,:,2) = G(og);

og2(:,:,3) = B(og);

og2 = im2uint8(og2);

imshow(og2);

**Screenshot:**

****

**(xi) Mean of an image**

**Algorithm:**

Step1: Clear all the text from the command window

Step2: Read the image lena

Step3: Convert the rgb image to grey

Step4: Find the sum

Step5: find the number of pixels

Step6: Compute the mean by dividing the sum with the number of pixels.

**Code:**

clc

I=imread('lena.png');

I=rgb2gray(I);

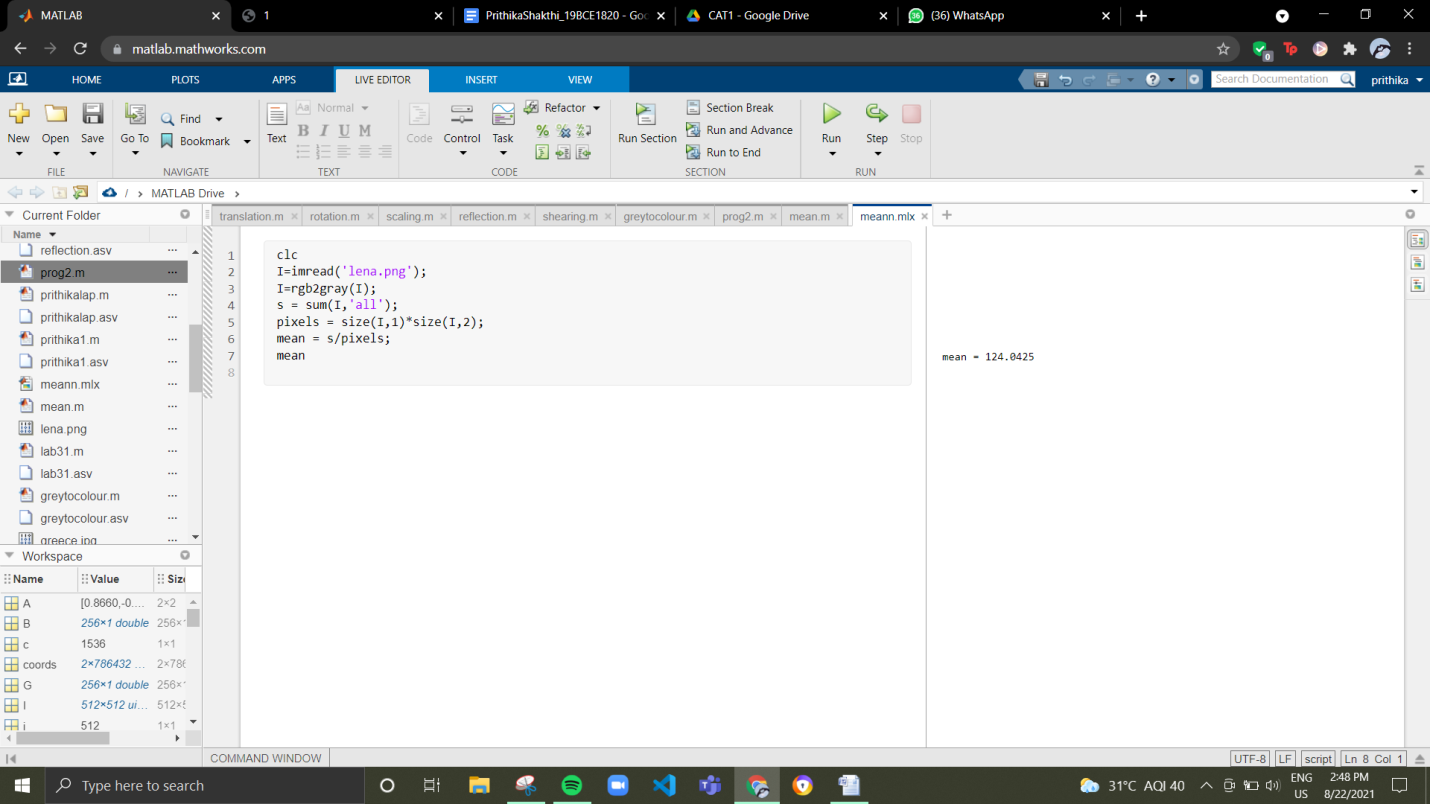
s = sum(I,'all');

pixels = size(I,1)\*size(I,2);

mean = s/pixels;

mean

**Screenshot:**

****

**(xii) Median of an image**

**Algorithm:**

Step1: Clear all the text from the command window

Step2: Read the image lena.

**Code:**

img=imread('lena.png');

x=median(img(:)) ;

y=sort(img(:));

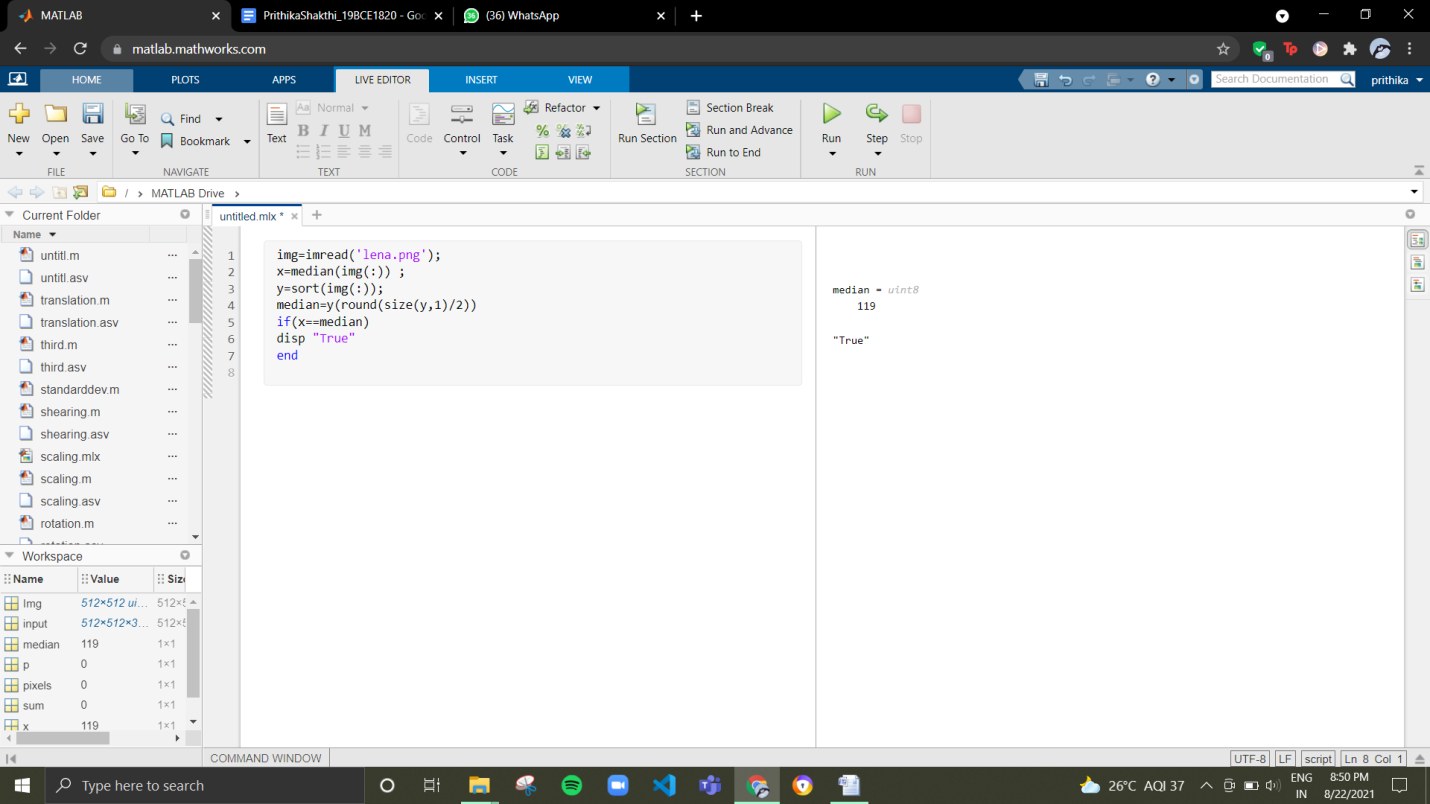
median=y(round(size(y,1)/2))

if(x==median)

disp "True"

end

**Screenshot:**

****

**(xiii) Standard deviation**

**Algorithm:**

Step1: Clear all the text from the command window

Step2: Read the picture lena and store it in I

Step3: Convert the rgb image to grey

Step4: Initialize Sum, P and Pixel value to 0

Step5: Create a for loop and for r=1:size(I,1)

Step6: Create a for loop inside and check for c=1:size(I,2)

Step7: Now perform calculation for p=p+(I(r,c,1)^2)

Step8: Compute the sum

Step9: Initialise the value of pixel

Step10: Close both the for loops

Step11: Compute the mean by dividing sum by pixels

Step12: Compute the standard deviation using the formula std = sqrt(p/pixels-mean^2)

**Code:**

**clc**

**I=imread('lena.png');**

**I=im2double(rgb2gray(I));**

**sum = 0;**

**p = 0;**

**pixels = 0;**

**for r = 1:size(I, 1)**

**for c = 1:size(I, 2)**

**p = p+(I(r,c,1)^2);**

**sum = sum + I(r,c,1);**

**pixels = pixels+1;**

**end**

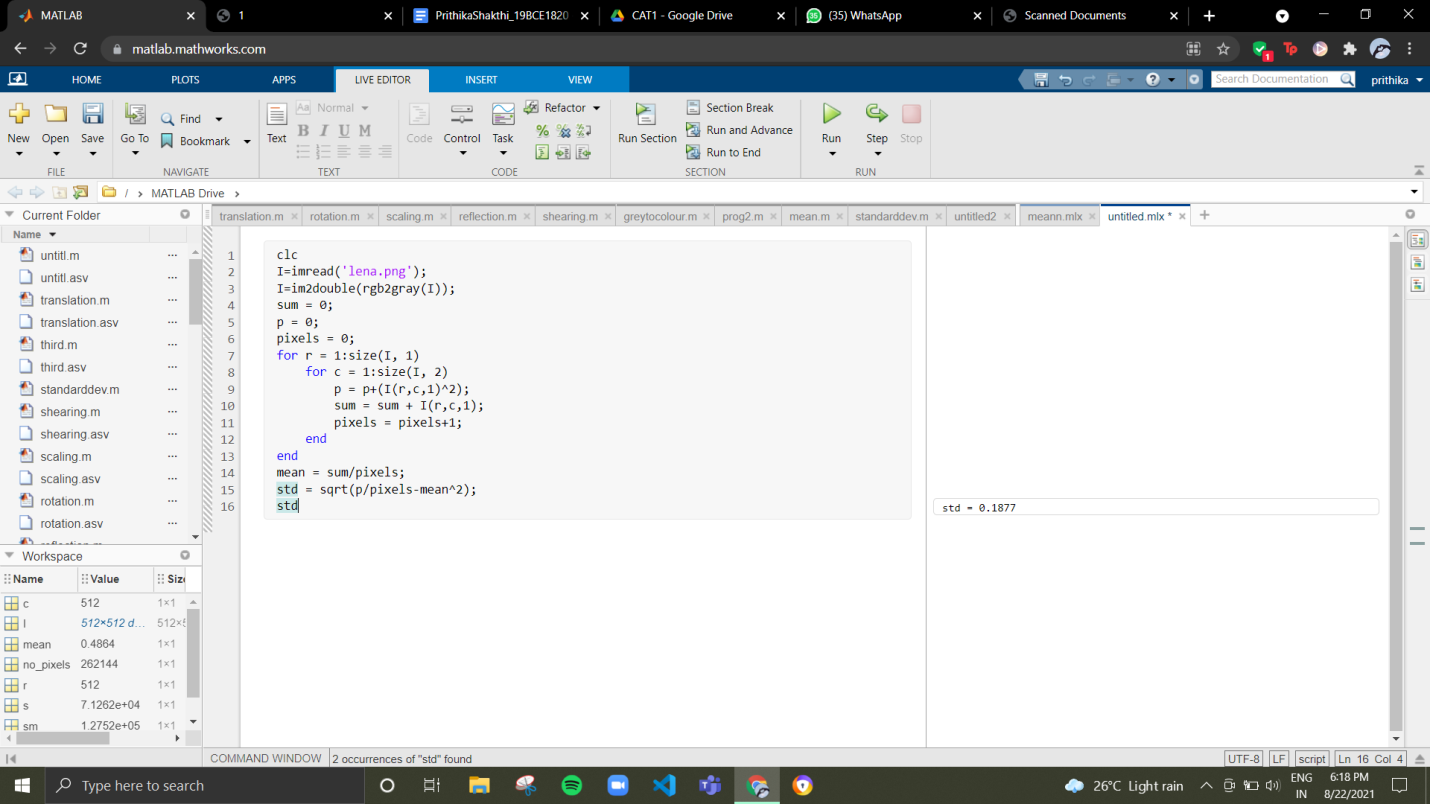
**end**

**mean = sum/pixels;**

**std = sqrt(p/pixels-mean^2);**

**std**

**Screenshot:**

****

**(xiv) Variance of an image**

**Algorithm:**

Step1: Read the Image lena and store it in Img

Step2: Convert the rgb image to grayscale

Step3: Initialize Sum, p and number of pixels to 0

Step4: The sum of the square values obtained is divided by the size of the standard image to find the variance

Step5: The mean value is subtracted from each individual element of the matrix and it is then squared

**Code:**

**Img=imread('lena.png');**

**Img=rgb2gray(Img);**

**sum = 0;**

**p = 0;**

**pixels = 0;**

**for r = 1:size(I, 1)**

**for c = 1:size(I, 2)**

**p = p+(I(r,c,1)^2);**

**sm = sm + I(r,c,1);**

**pixels = pixels+1;**

**end**

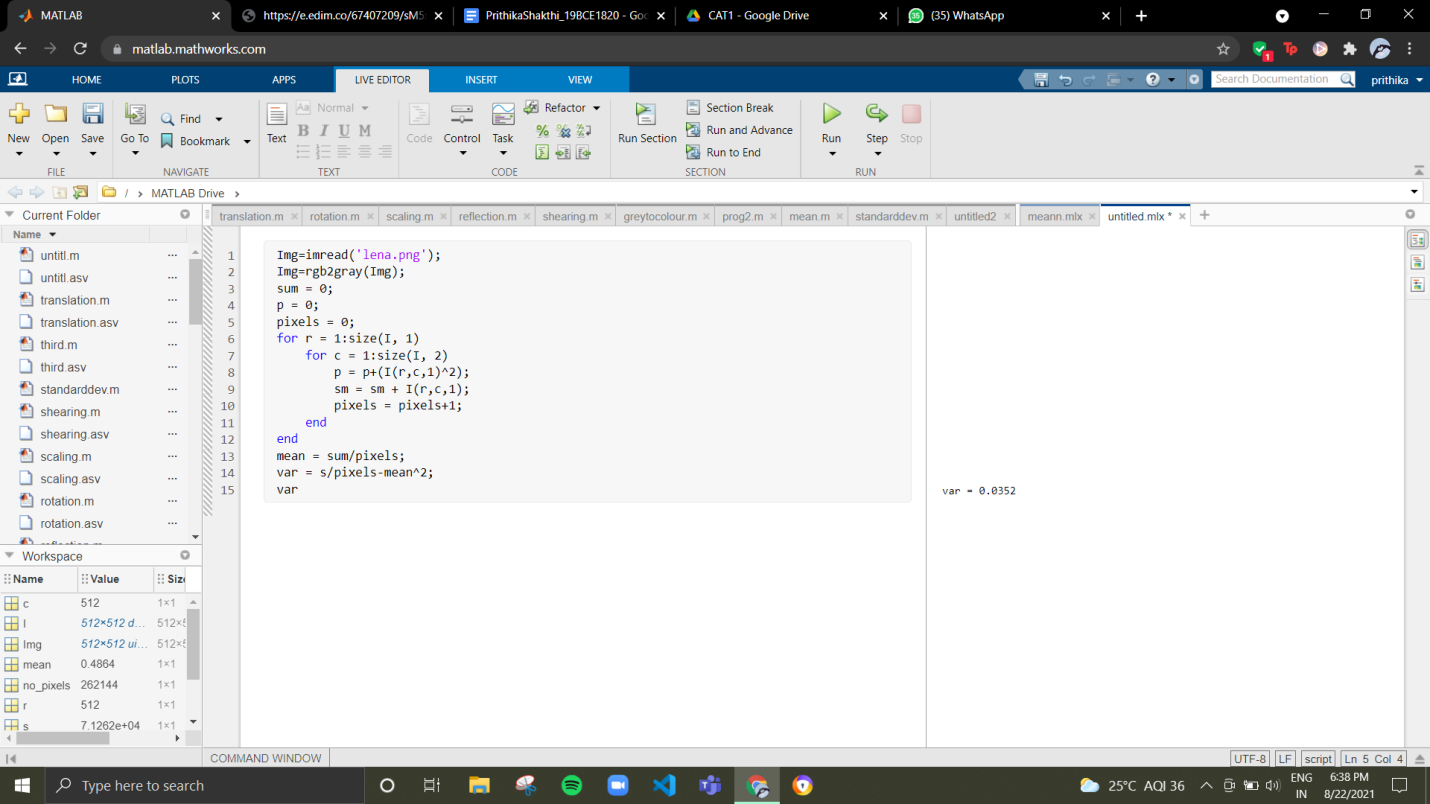
**end**

**mean = sum/pixels;**

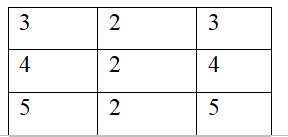
**var = s/pixels-mean^2;**

**var**

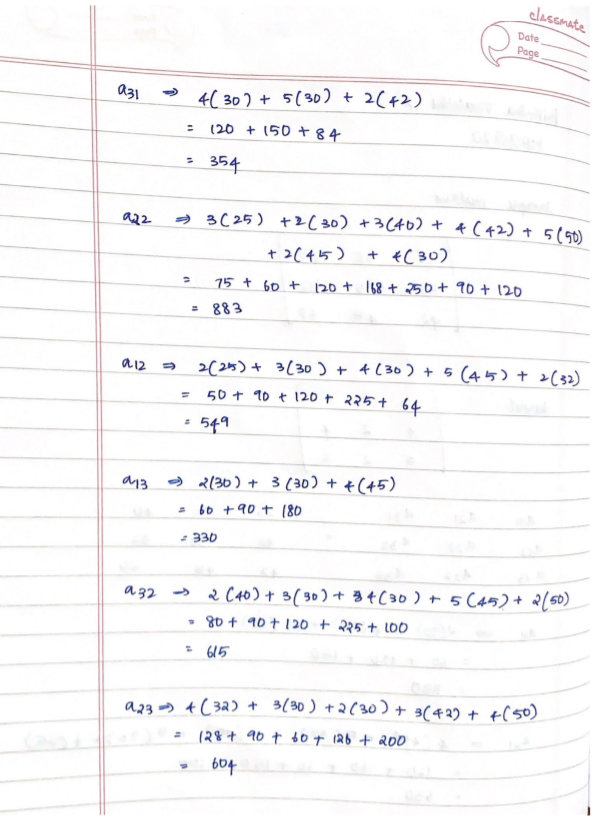
**Screenshot:**

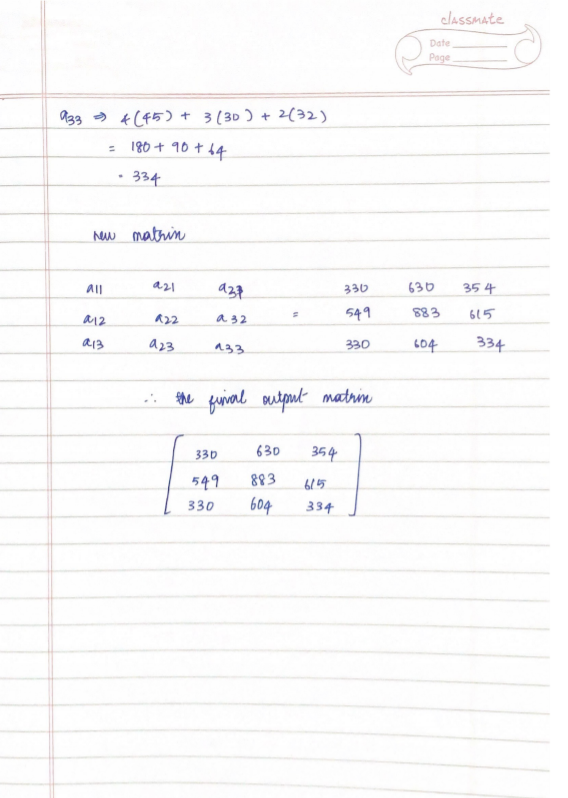
****

1. **Read any sample 3x3 image matrix, convolve it using the given matrix as shown below and find the convolution values.**

****

****

****

****