Practical Set-3

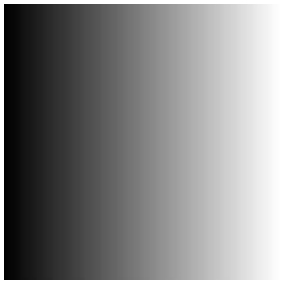
Tool: Matlab

- Consider the following matrix

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 2 & \dots & 255 \\ 0 & 1 & 2 & \dots & 255 \\ \vdots & & & & \\ 0 & 1 & 2 & \dots & 255 \end{bmatrix}$$

Generate the above matrix and show the equivalent grayscale image.

i.e.



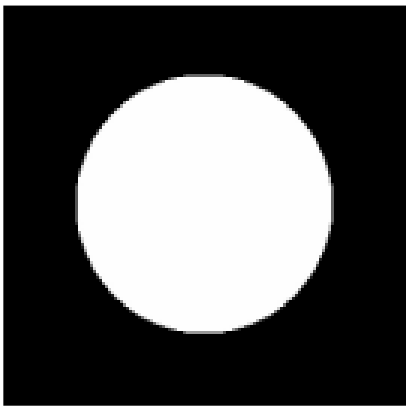
.

- Generate the following matrix of size 256 x 256. (It will generate circle centered at (128,128) and radius=80 pixels).

$$B(i,j) = \begin{cases} 255 & \text{if } \sqrt{(i-128)^2 + (j-128)^2} < 80 \\ 0 & \text{otherwise} \end{cases}$$

Generate the equivalent grayscale image.

i.e.

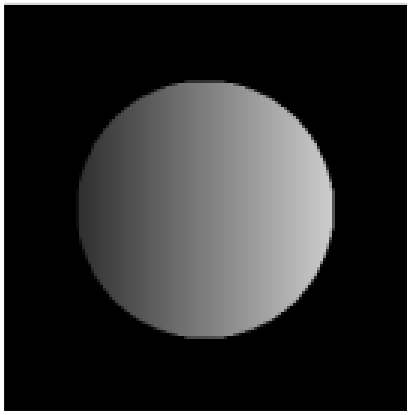


- Generate the matrix "C" of size 256 x 256, where,

C (i, j) = A (i, j) x B (i, j).

Generate the equivalent grayscale image.

i.e.



- Make a matlab function “subsample (image,factor)”, where image is the image to be sub sampled and factor specifies the sub sampling factor (if factor is 1, 1024 x 1024 image should be sub-sampled to 512 x 512, if it is 2 then it should be sub sampled to 256 x 256). Function must display the sub-sampled image at the original size.

- Demonstrate the effect of false contouring due to reduction in number of gray levels.
  (a) Use “imshow (image, number of gray levels)”.
  (b) Use “imshow()”, but do not use second argument specifying the number of gray levels.

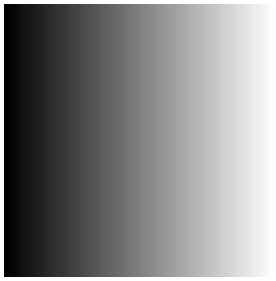
Solution:

1)Consider the following matrix

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 2 & \dots & 255 \\ 0 & 1 & 2 & \dots & 255 \\ \vdots & & & & \\ 0 & 1 & 2 & \dots & 255 \end{bmatrix}$$

Generate the above matrix and show the equitant grayscale image.

i.e.

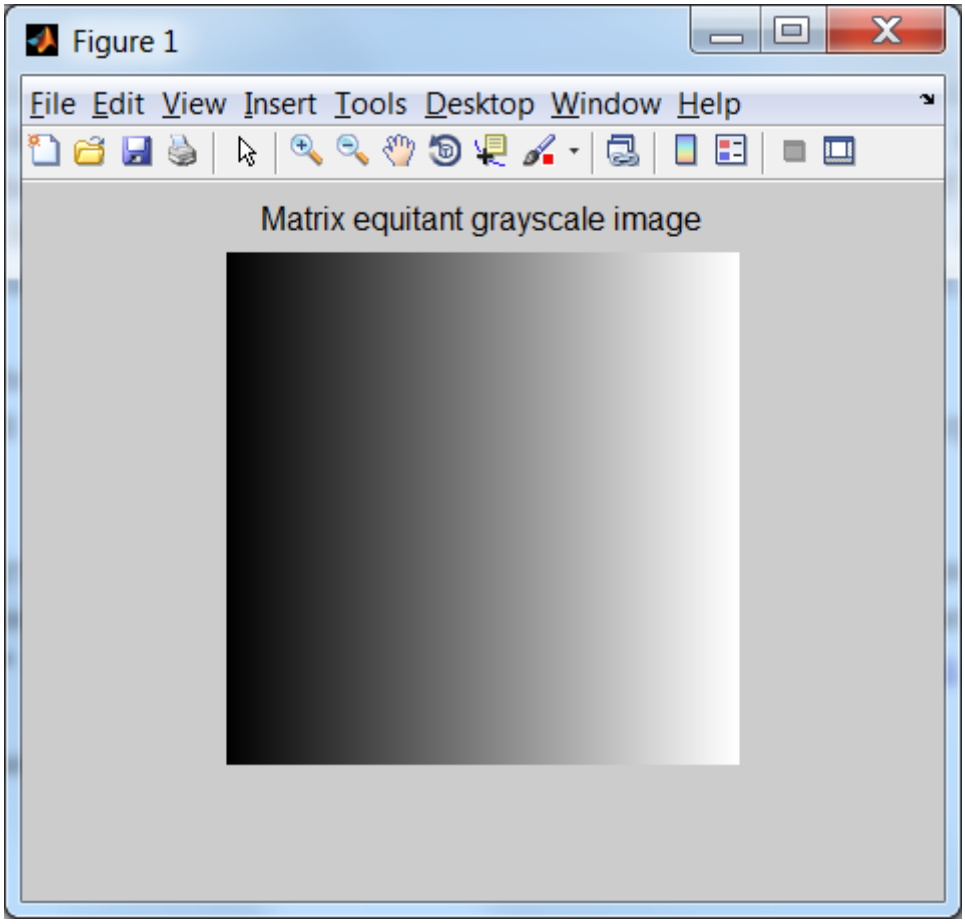
. 

Program:

```
function parc3one()
% Generate the matrix and show equitant grayscale image

% Function : show matrix equitant grayscale image
% Description : This function is use for the show matrix equitant grayscale
% image.First generate the 256*256 matrix for image.
a=zeros(256,256,'uint8');
for i=1:256
    for j=1:256
        a(i,j)=j;
    end
end
imshow(a),title('Matrix equitant grayscale image')
end
```

output:

>>parc3

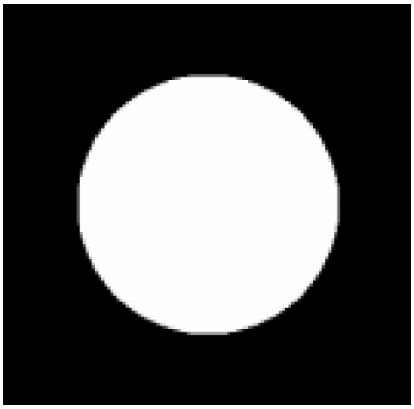


2)Generate the following matrix of size 256 x 256. (It will generate circle centered at (128,128) and radius=80 pixels).

$$B(i,j) = \begin{cases} 255 & \text{if } \sqrt{(i-128)^2 + (j-128)^2} < 80 \\ 0 & \text{otherwise} \end{cases}$$

Generate the equivalent grayscale image.

i.e.

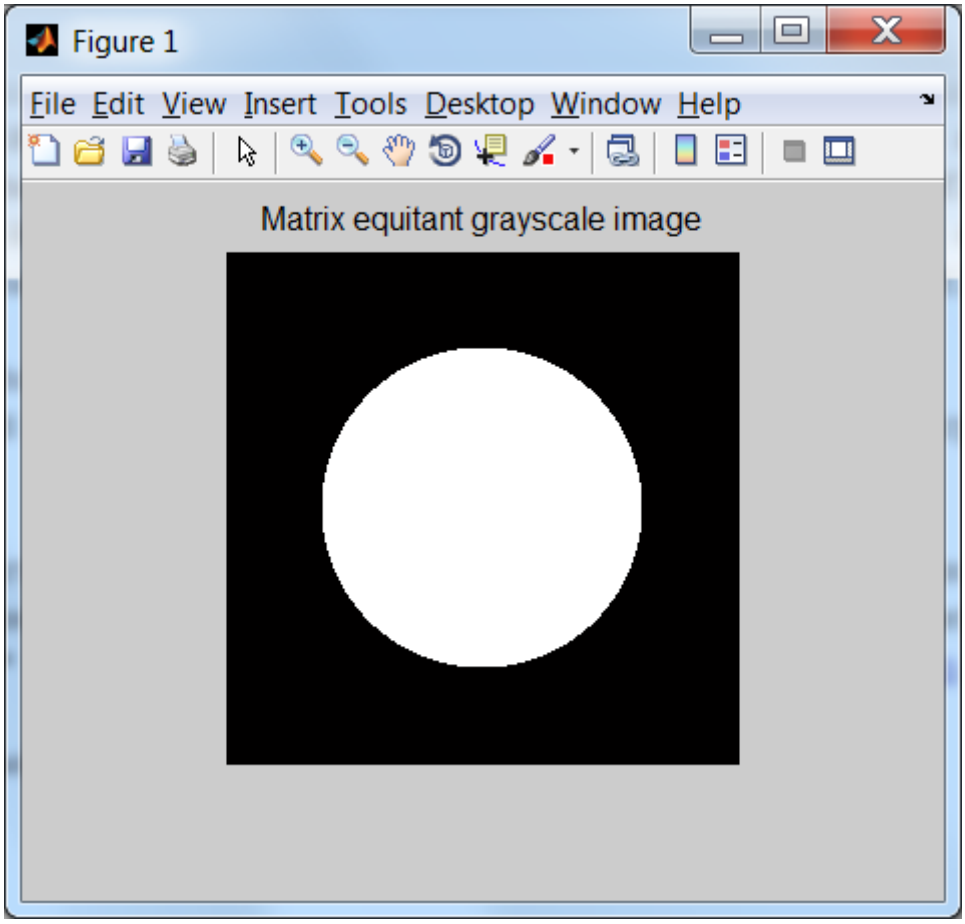


Program:

```
function parc3two()
% Generate the matrix and show equitant grayscale image

% Function : show matrix equitant grayscale image
% Description : This function is use for the show matrix equitant grayscale
% image.First generate the 256*256 matrix for image.It will generate circle
% centered at (128,128) and radius=80 pixels).
% b(i,j)=255 if ((i-128)^2+(j-128)^2<80)^1/2 and 0 otherwise
b=zeros(256,256,'uint8');
for i=1:256
    for j=1:256
        if((sqrt(power(i-128,2)+power(j-128,2)))<80)
            b(i,j)=256;
        else
            b(i,j)=0;
        end
    end
end
    imshow(b),title('Matrix equitant grayscale image');
end
```

output:

```
>> parc3two
```

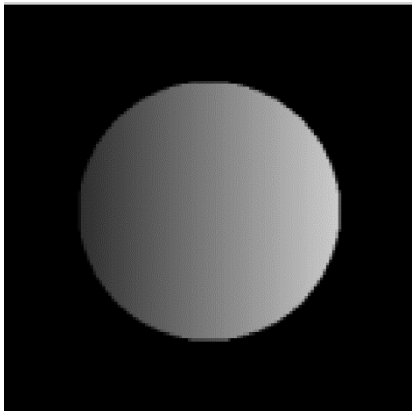


- Generate the matrix “C” of size 256 x 256, where,

  C (i, j) = A (i, j) x B (i, j).

  Generate the equivalent grayscale image.

  i.e.



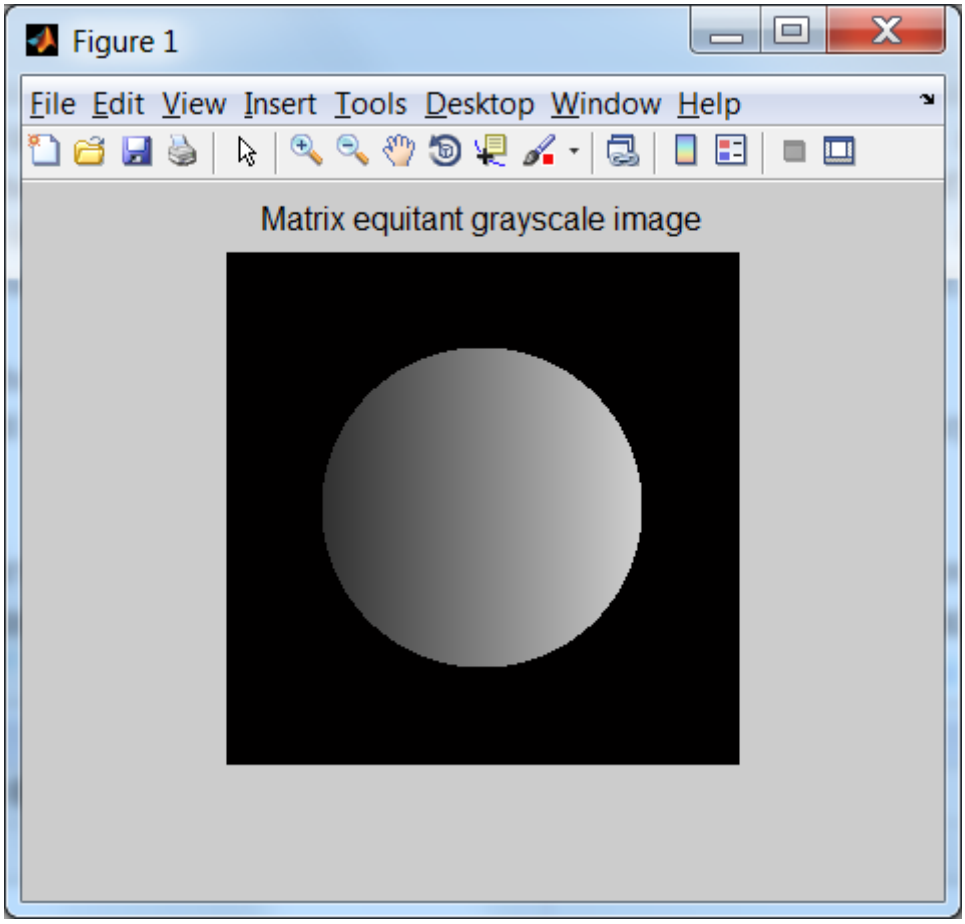
Program:

```
function matrix3()
% Generate the matrix and show equitant grayscale image

% Function : show matrix equitant grayscale image
```

```
% Description : This function is use for the show matrix equitant grayscale
% image.First generate the 256*256 matrix for image.
% here we use c(i,j)=a(i,j)*b(i,j)

a=zeros(256,256,'uint8');
for i=1:256
    for j=1:256
        a(i,j)=j;
    end
end
b=zeros(256,256,'uint8');
for i=1:256
    for j=1:256
        if((sqrt(power(i-128,2)+power(j-128,2)))<80)
            b(i,j)=256;
        else
            b(i,j)=0;
        end
    end
end
c=zeros(256,256,'uint8');
for i=1:256
    for j=1:256
        d=im2double(a);
            c(i,j)=d(i,j)*b(i,j);
        end
    end
    imshow(c);
end
```

Output:

```
>>matrix3
```

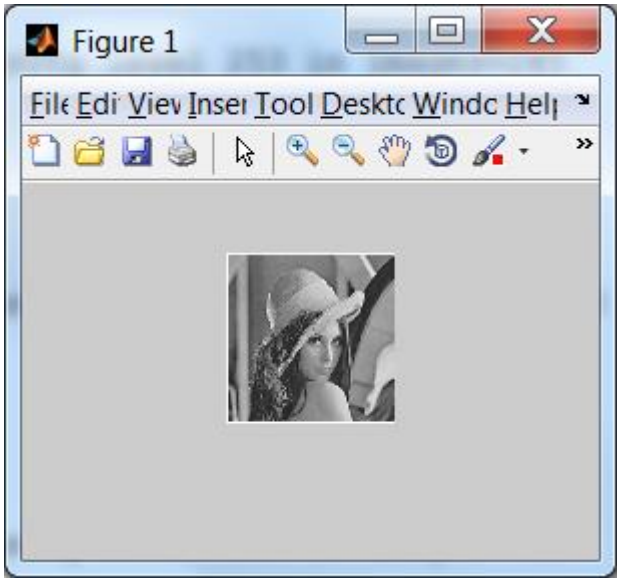


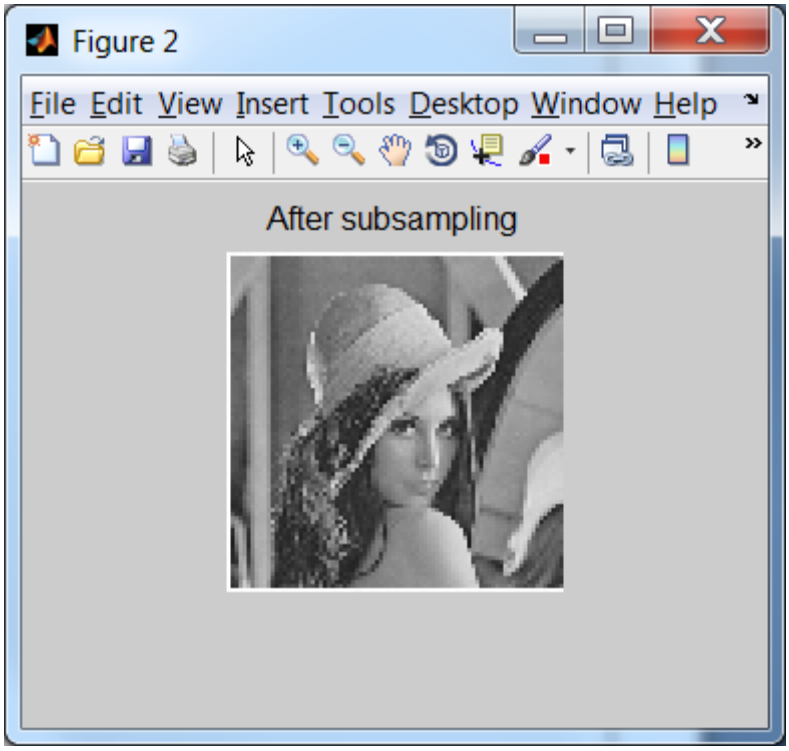
4)Make a matlab function “subsample (image,factor)”, where image is the image to be sub sampled and factor specifies the sub sampling factor (if factor is 1, 1024 x 1024 image should be sub-sampled to 512 x 512, if it is 2 then it should be sub sampled to 256 x 256). Function must display the sub-sampled image at the original size.

```
function parc3four(image,factor)

% Make a matlab function “subsample (image,factor)”, where image is the
image
% to be sub sampled and factor specifies the sub sampling factor (if factor
% is 1, 1024 x 1024 image should be sub-sampled to 512 x 512, if it is 2
then
% it should be sub sampled to 256 x 256). Function must display the
% sub-sampled image at the original size..

% Function:subsampling function

a=imread(image);
for i=1:factor
    a=a(:,1:2:end);
    a=a(1:2:end,:);
end
imshow(a);
[x y z]=size(a);
for j=1:factor
```

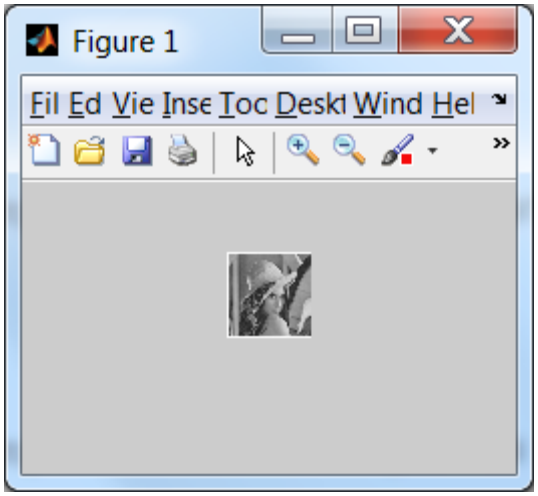
```
    k=1;
    for i=1:y
        b(:,k)=a(:,i);
        b(:,k+1)=a(:,i);
        k=k+2;
    end
    a=b;
    [x y z]=size(a);
    k=1;
    for i=1:x
        b(k,:)=a(i,:);
        b(k+1,:)=a(i,:);
        k=k+2;
    end
    a=b;
    [x y z]=size(a);
End
figure;
imshow(b),,title('After subsampling');
end
```

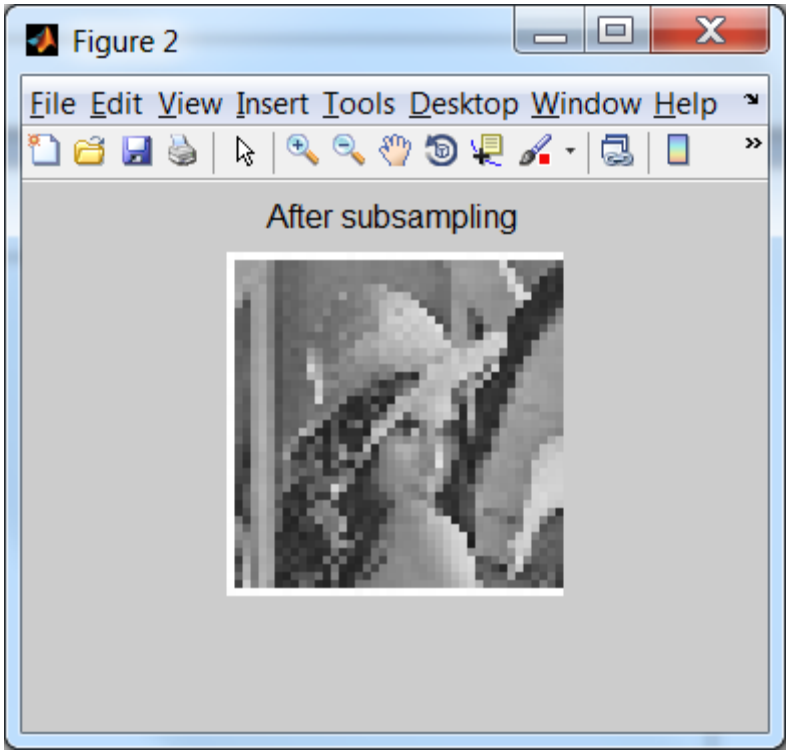
>> parc3four('img3.png',1)





>> parc3four('img3.png',2)





5)Demonstrate the effect of false contouring due to reduction in number of gray levels.

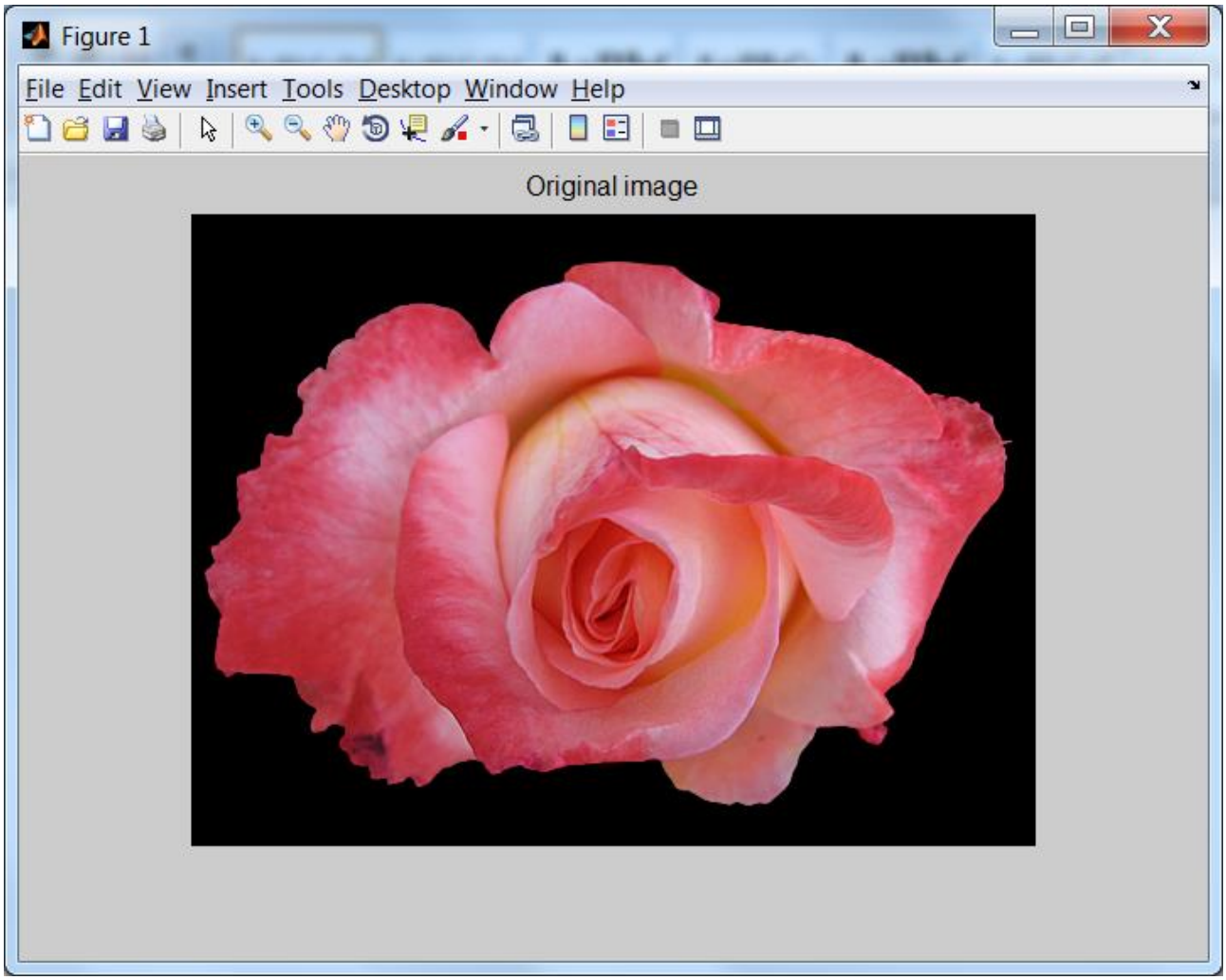
a. Use “imshow()”, but do not use second argument specifying the number of gray levels.

```
function parc3five(image1)
% Demonstrate the effect of false contouring due to reduction in number o
% gray levels.
% (a)Use “imshow()”, but do not use second argument specifying the
% number of gray levels.

% Function:effect of false contouring due to reduction in number
% of gray levels
% Description:This function is use for show the effect of the false
% contouring without use second argument specifying the
% number of gray levels.
imshow(image1),title('original image');
title('Original image');
img=imread(image1);
[r c z]=size(img);
if z == 3
    b=rgb2gray(img);
else
    b=image1;
end
for i=1:r
    for j=1:c
        if (b(i,j) >= 0 && b(i,j) < 64)
            b(i,j) = 64;
        elseif (b(i,j) >= 64 && b(i,j) < 128)
            b(i,j)=128;
        elseif (b(i,j) >=128 && b(i,j) < 192)
            b(i,j)=192;
        else
            b(i,j)=256;
```

```
        end
    end
end
figure;
imshow(b),title('After False contouring');
end
```

output:

>> parc3five('rose1.png')



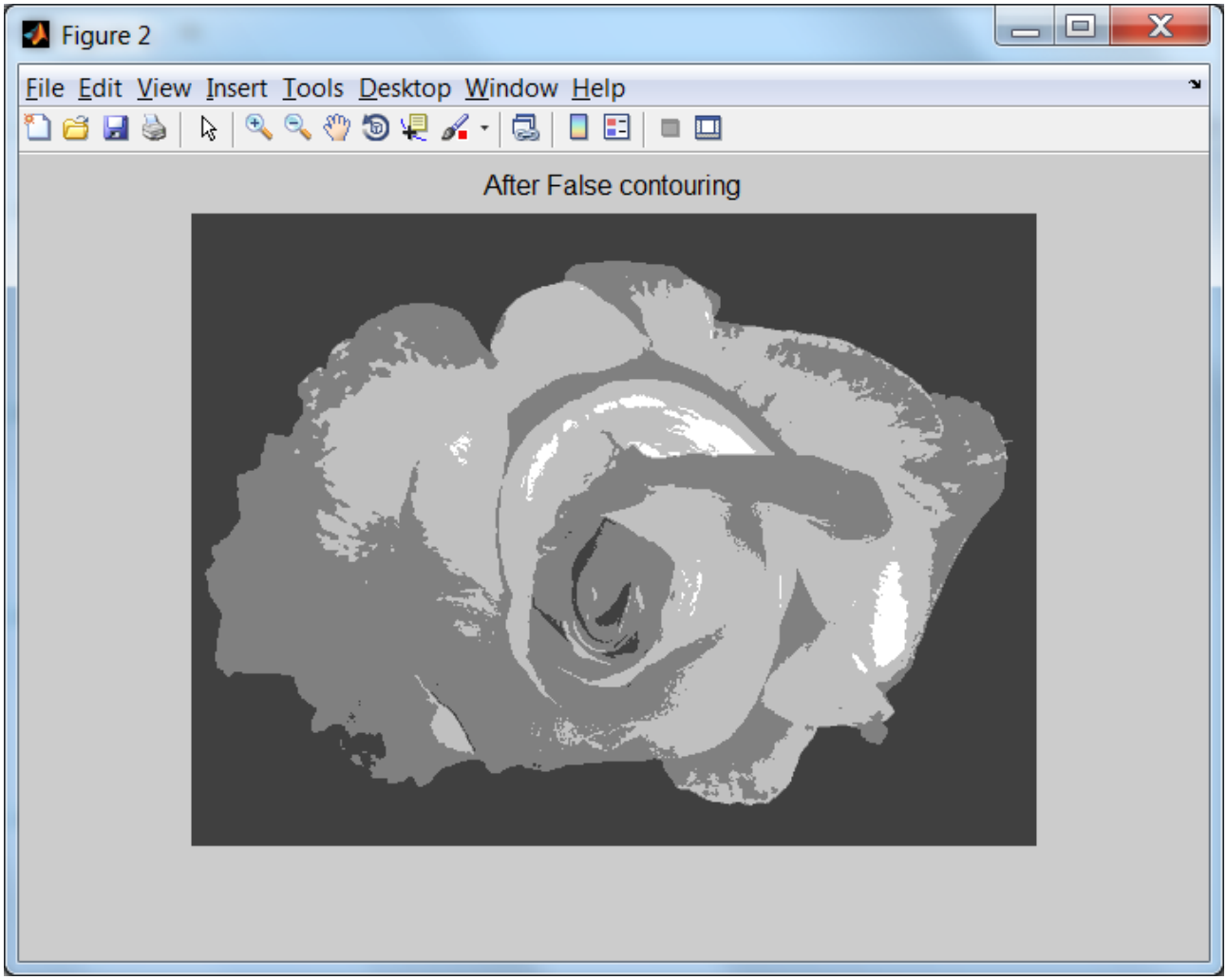


Figure 2
File Edit View Insert Tools Desktop Window Help
After False contouring

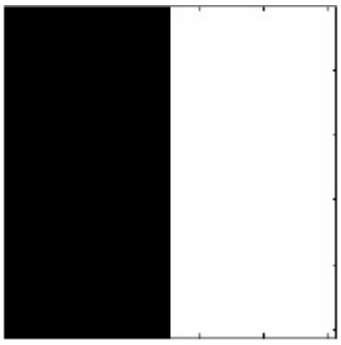
Practical Set-4

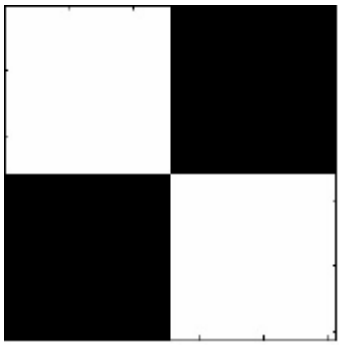
AIM:

Tool: Matlab

1. Generate histograms for the following images. (1) Display bar charts. (2) Display number of pixels belonging to each intensity level in images, in terms of statement. (3) Generate normalize histograms. (4) Invert images and generate their histogram. (File: Original Images, their bar charts, normalize bar charts, Inverted images and their bar charts and statements as asked above)

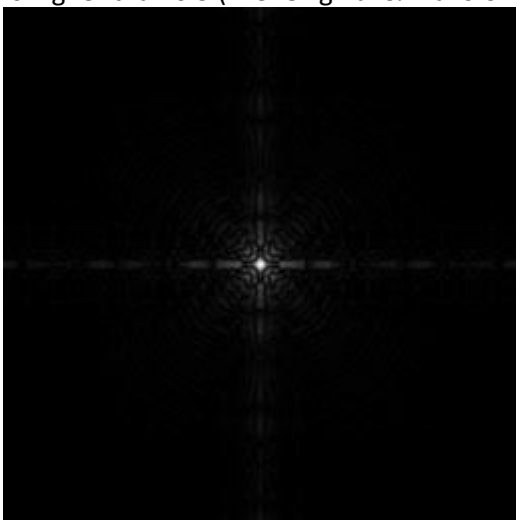
(a) (b)





2. Generate histogram for the following image. Invert the image and generate the histogram. Rotate the image by 180 degree clockwise and generate the histogram. (File: Original image, Inverted image, Rotated Image and their histograms.)

3. Apply log transformation on the following image. Show the transform image and part of the image whose intensity is higher than 0.5 (File: Original & Transform image, conclusion).



4. Apply power law transformation on the following image with c=1 and gamma=0.6, 0.4, 0.3. (File: Original image & Transform images, conclusion).

5. Apply power law transformation on the following image with c=1 and gamma=3, 4, 5. (File: Original image & Transform images, conclusion).

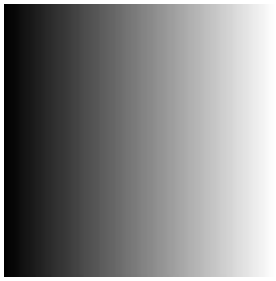
Solution:

1)Consider the following matrix

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 2 & \dots & 255 \\ 0 & 1 & 2 & \dots & 255 \\ \vdots & & & & \\ 0 & 1 & 2 & \dots & 255 \end{bmatrix}$$

Generate the above matrix and show the equitant grayscale image.

i.e.



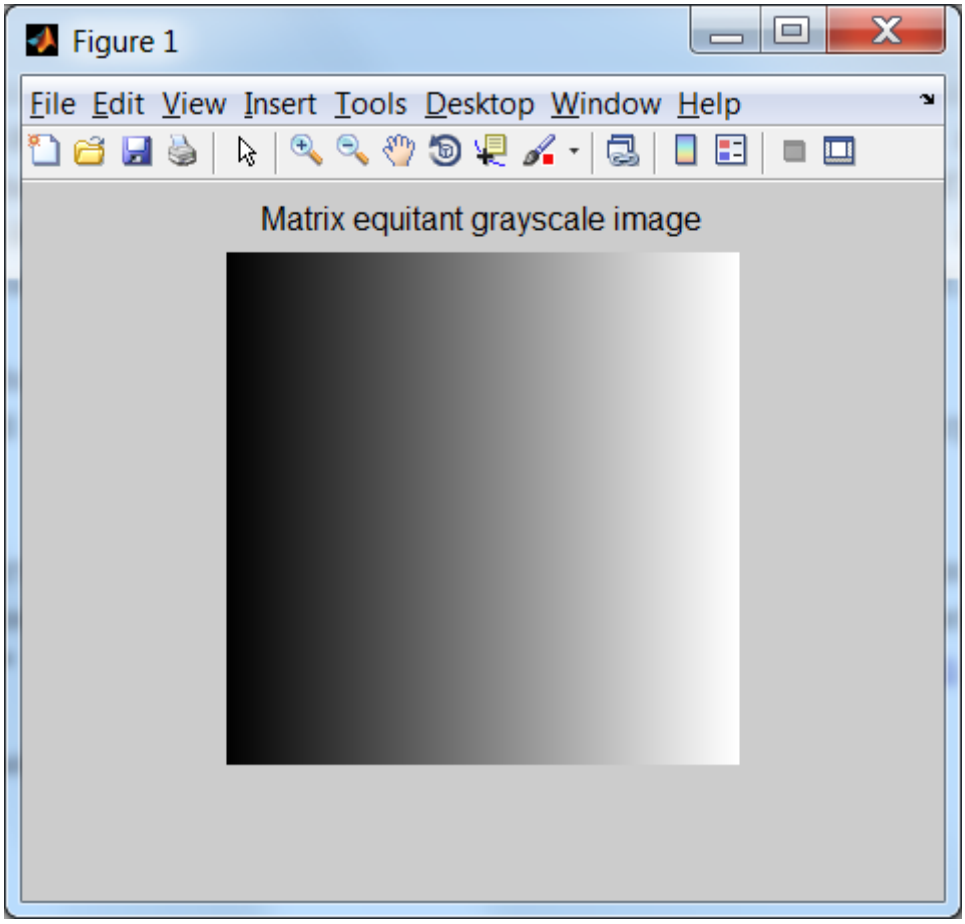
.

Program:

```
function parc3one()
% Generate the matrix and show equitant grayscale image

% Function : show matrix equitant grayscale image
% Description : This function is use for the show matrix equitant grayscale
% image.First generate the 256*256 matrix for image.
a=zeros(256,256,'uint8');
for i=1:256
    for j=1:256
        a(i,j)=j;
    end
end
imshow(a),title('Matrix equitant grayscale image')
end
```

output:

>>parc3

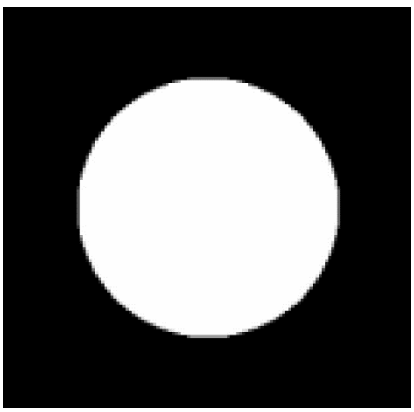


2)Generate the following matrix of size 256 x 256. (It will generate circle centered at (128,128) and radius=80 pixels).

$$B(i,j) = \begin{cases} 255 & \text{if } \sqrt{(i-128)^2 + (j-128)^2} < 80 \\ 0 & \text{otherwise} \end{cases}$$

Generate the equivalent grayscale image.

i.e.



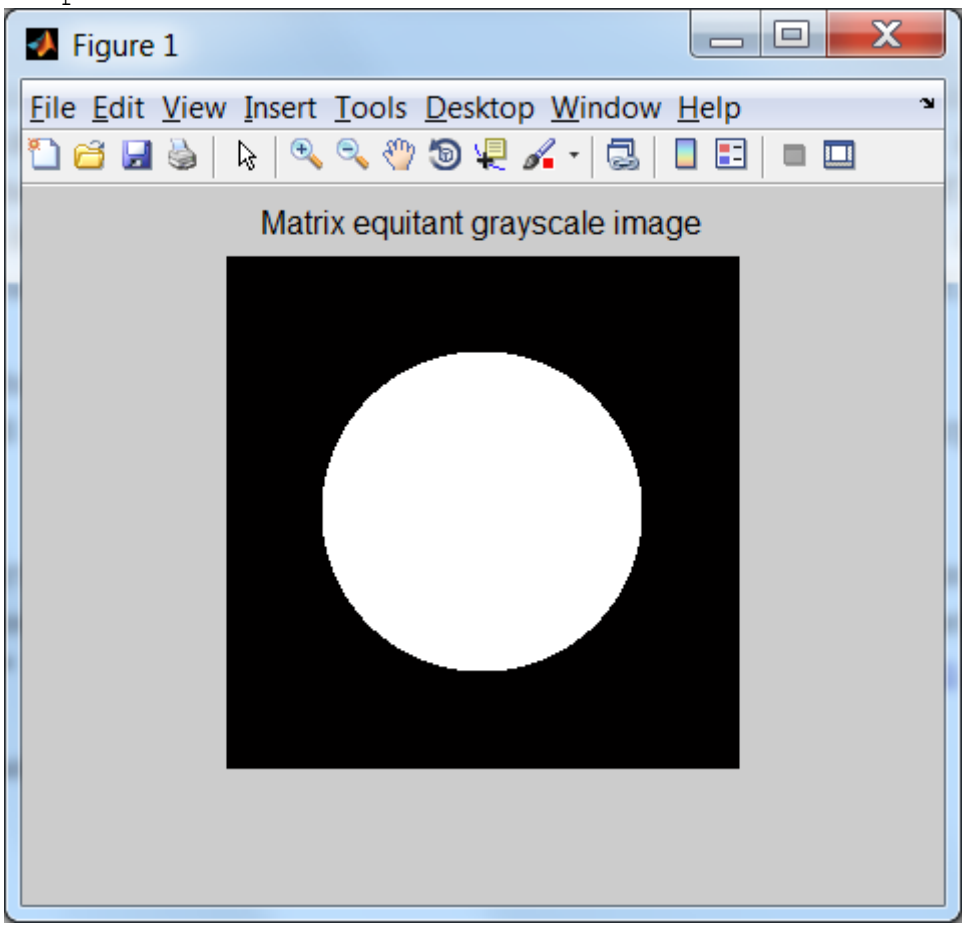
Program:

```
function parc3two()
% Generate the matrix and show equitant grayscale image
```

```
% Function : show matrix equitant grayscale image
% Description : This function is use for the show matrix equitant grayscale
% image.First generate the 256*256 matrix for image.It will generate circle
% centered at (128,128) and radius=80 pixels).
% b(i,j)=255 if ((i-128)^2+(j-128)^2<80)^1/2 and 0 otherwise
b=zeros(256,256,'uint8');
for i=1:256
    for j=1:256
        if((sqrt(power(i-128,2)+power(j-128,2)))<80)
            b(i,j)=256;
        else
            b(i,j)=0;
        end
    end
end
    imshow(b),title('Matrix equitant grayscale image');
end
```

output:

```
>> parc3two
```

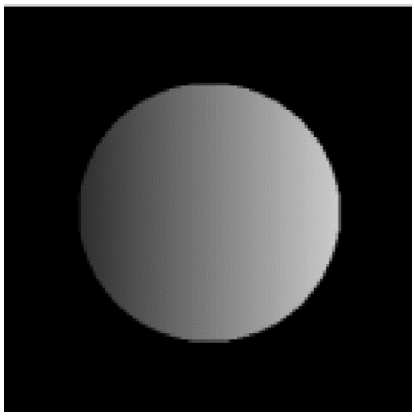


- Generate the matrix “C” of size 256 x 256, where,

  C (i, j) = A (i, j) x B (i, j).

  Generate the equivalent grayscale image.

  i.e.

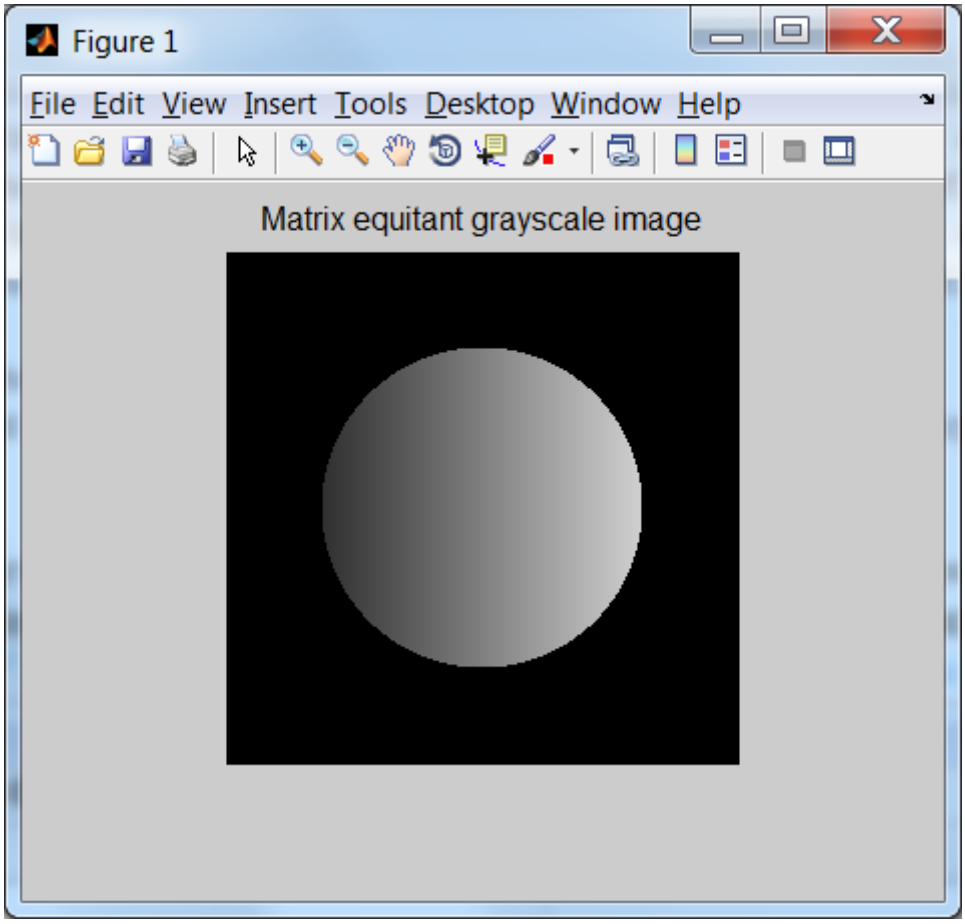


Program:

```
function matrix3()
% Generate the matrix and show equitant grayscale image

% Function : show matrix equitant grayscale image
% Description : This function is use for the show matrix equitant grayscale
% image.First generate the 256*256 matrix for image.
% here we use c(i,j)=a(i,j)*b(i,j)

a=zeros(256,256,'uint8');
for i=1:256
    for j=1:256
        a(i,j)=j;
    end
end
b=zeros(256,256,'uint8');
for i=1:256
    for j=1:256
        if((sqrt(power(i-128,2)+power(j-128,2)))<80)
            b(i,j)=256;
        else
            b(i,j)=0;
        end
    end
end
c=zeros(256,256,'uint8');
for i=1:256
    for j=1:256
        d=im2double(a);
            c(i,j)=d(i,j)*b(i,j);
        end
    end
    imshow(c);
end
```

Output:

```
>>matrix3
```

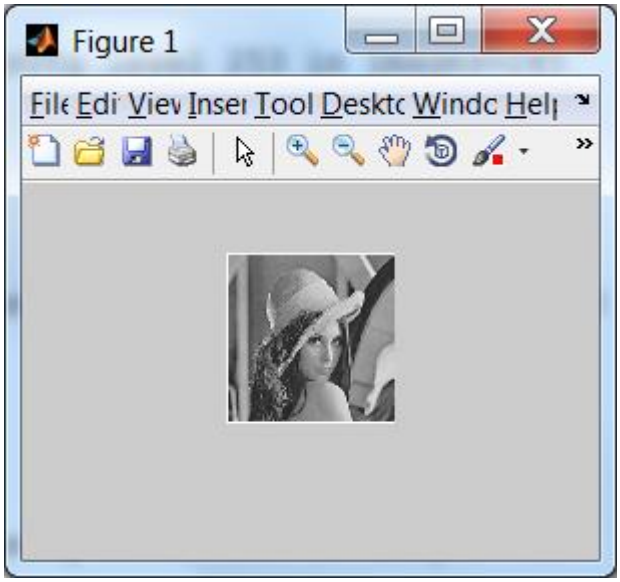


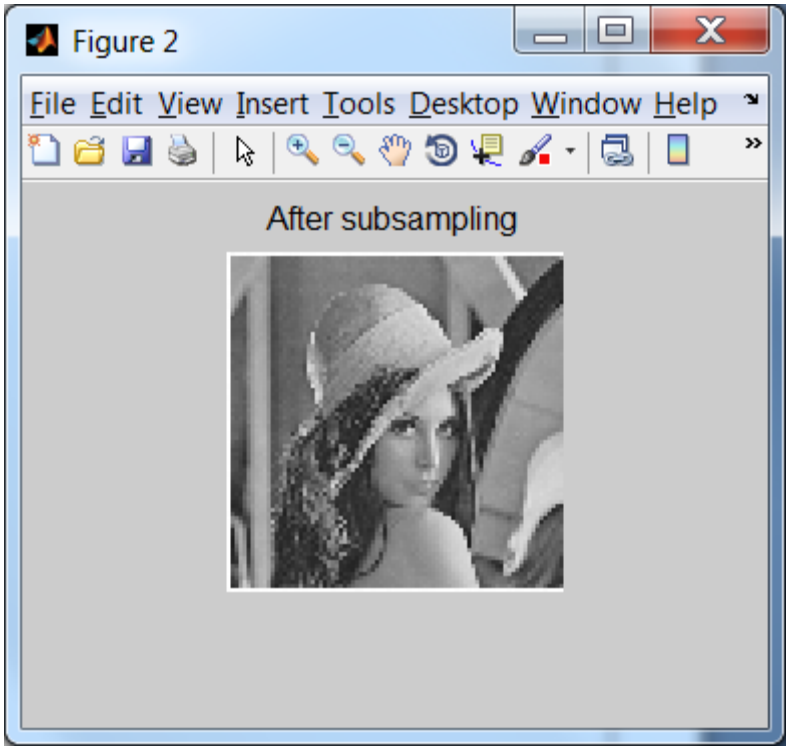
4)Make a matlab function “subsample (image,factor)”, where image is the image to be sub sampled and factor specifies the sub sampling factor (if factor is 1, 1024 x 1024 image should be sub-sampled to 512 x 512, if it is 2 then it should be sub sampled to 256 x 256). Function must display the sub-sampled image at the original size.

```
function parc3four(image,factor)

% Make a matlab function “subsample (image,factor)”, where image is the
image
% to be sub sampled and factor specifies the sub sampling factor (if factor
% is 1, 1024 x 1024 image should be sub-sampled to 512 x 512, if it is 2
then
% it should be sub sampled to 256 x 256). Function must display the
% sub-sampled image at the original size..

% Function:subsampling function

a=imread(image);
for i=1:factor
    a=a(:,1:2:end);
    a=a(1:2:end,:);
end
imshow(a);
[x y z]=size(a);
for j=1:factor
```

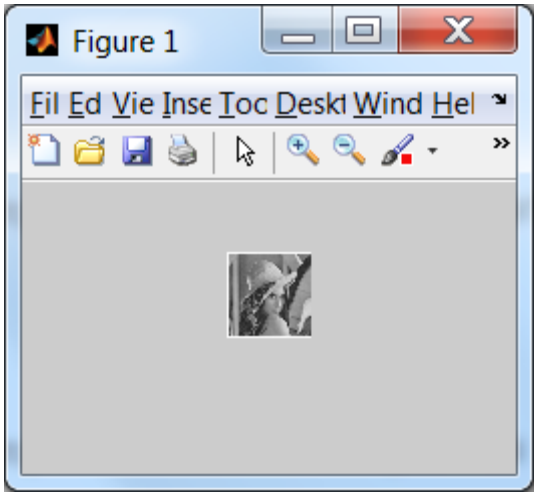
```
    k=1;
    for i=1:y
        b(:,k)=a(:,i);
        b(:,k+1)=a(:,i);
        k=k+2;
    end
    a=b;
    [x y z]=size(a);
    k=1;
    for i=1:x
        b(k,:)=a(i,:);
        b(k+1,:)=a(i,:);
        k=k+2;
    end
    a=b;
    [x y z]=size(a);
End
figure;
imshow(b),,title('After subsampling');
end
```

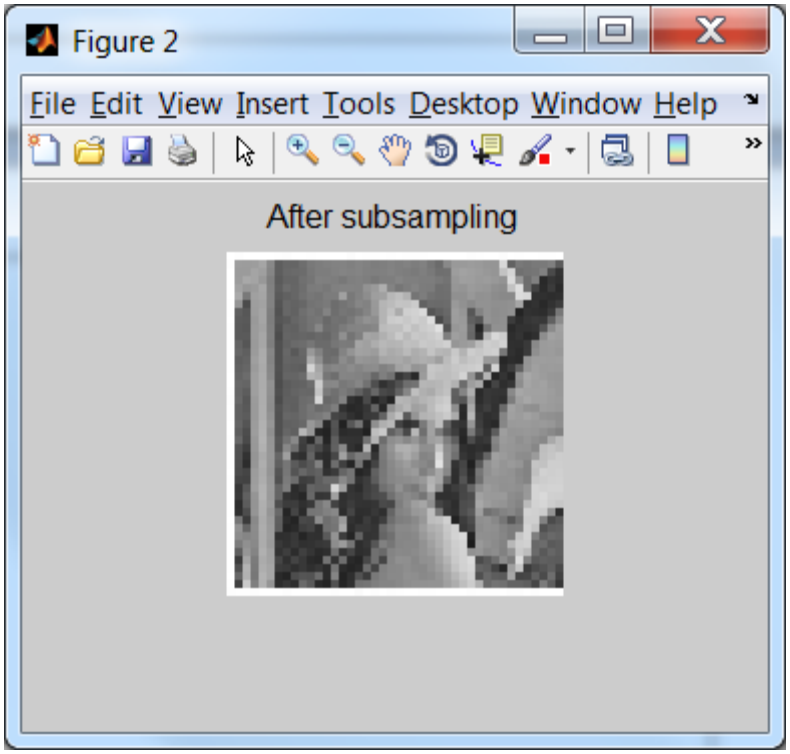
>> parc3four('img3.png',1)





>> parc3four('img3.png',2)





5)Demonstrate the effect of false contouring due to reduction in number of gray levels.

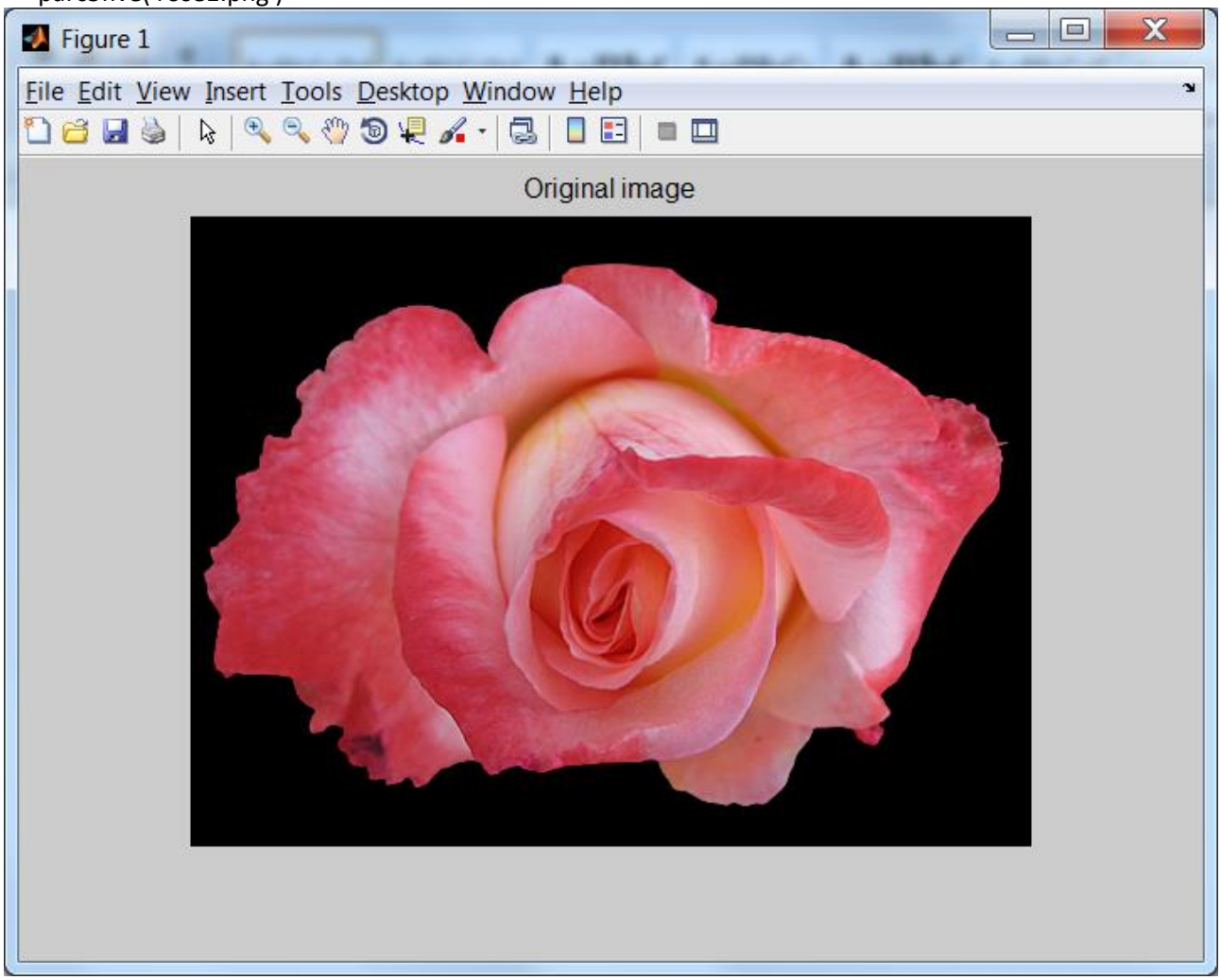
a. Use “imshow()”, but do not use second argument specifying the number of gray levels.

```
function parc3five(image1)
% Demonstrate the effect of false contouring due to reduction in number o
% gray levels.
% (a)Use “imshow()”, but do not use second argument specifying the
% number of gray levels.

% Function:effect of false contouring due to reduction in number
% of gray levels
% Description:This function is use for show the effect of the false
% contouring without use second argument specifying the
% number of gray levels.
imshow(image1),title('original image');
title('Original image');
img=imread(image1);
[r c z]=size(img);
if z == 3
    b=rgb2gray(img);
else
    b=image1;
end
for i=1:r
    for j=1:c
        if (b(i,j) >= 0 && b(i,j) < 64)
            b(i,j) = 64;
        elseif (b(i,j) >= 64 && b(i,j) < 128)
            b(i,j)=128;
        elseif (b(i,j) >=128 && b(i,j) < 192)
            b(i,j)=192;
        else
            b(i,j)=256;
```

```
        end
    end
end
figure;
imshow(b),title('After False contouring');
end
```

output:

>> parc3five('rose1.png')



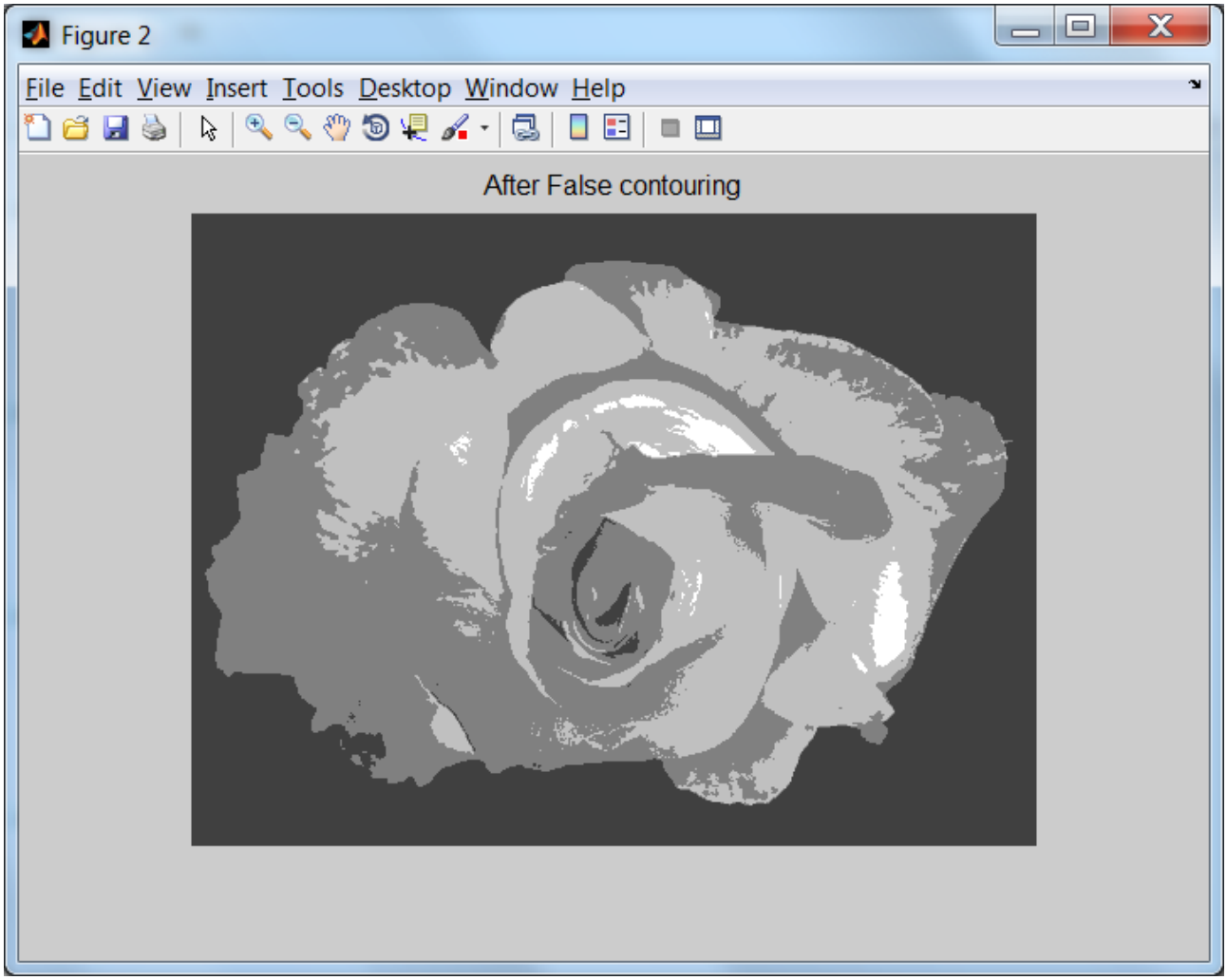
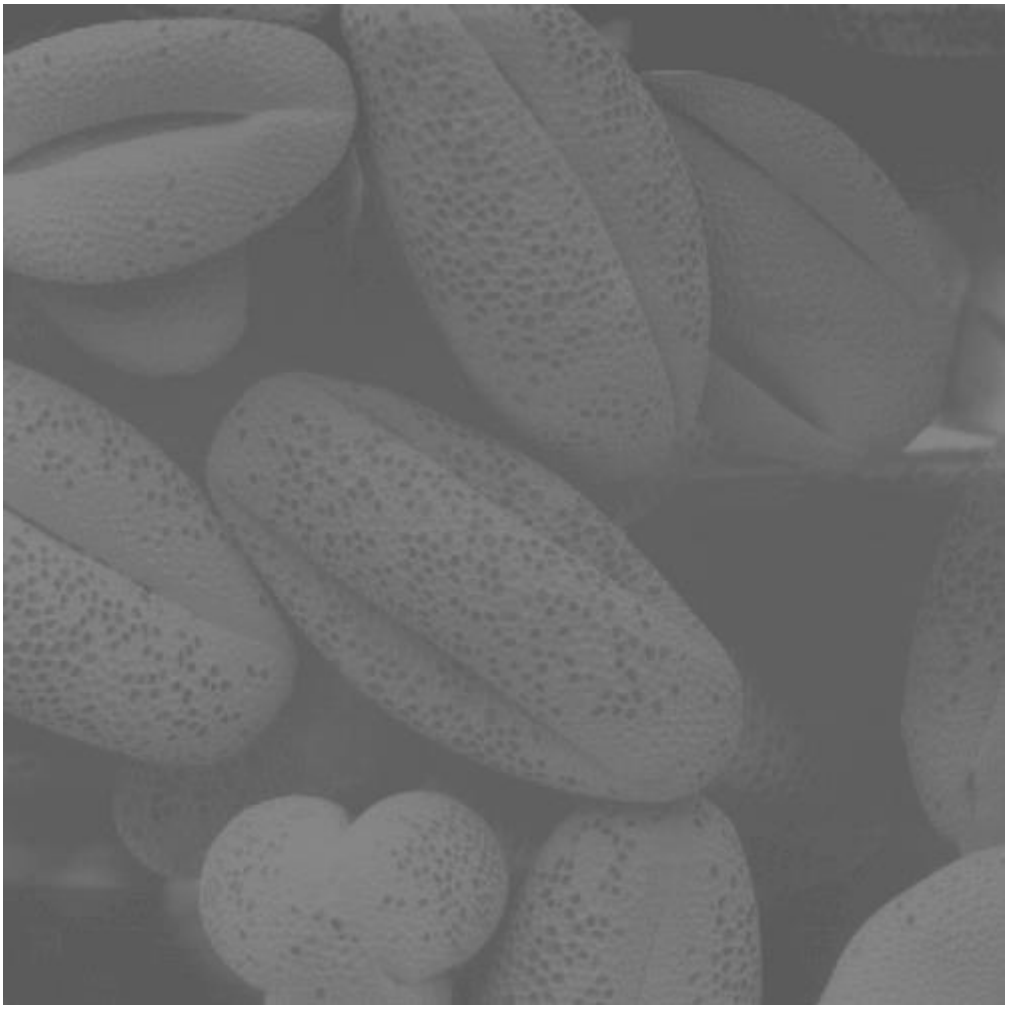


Figure 2
File Edit View Insert Tools Desktop Window Help
After False contouring

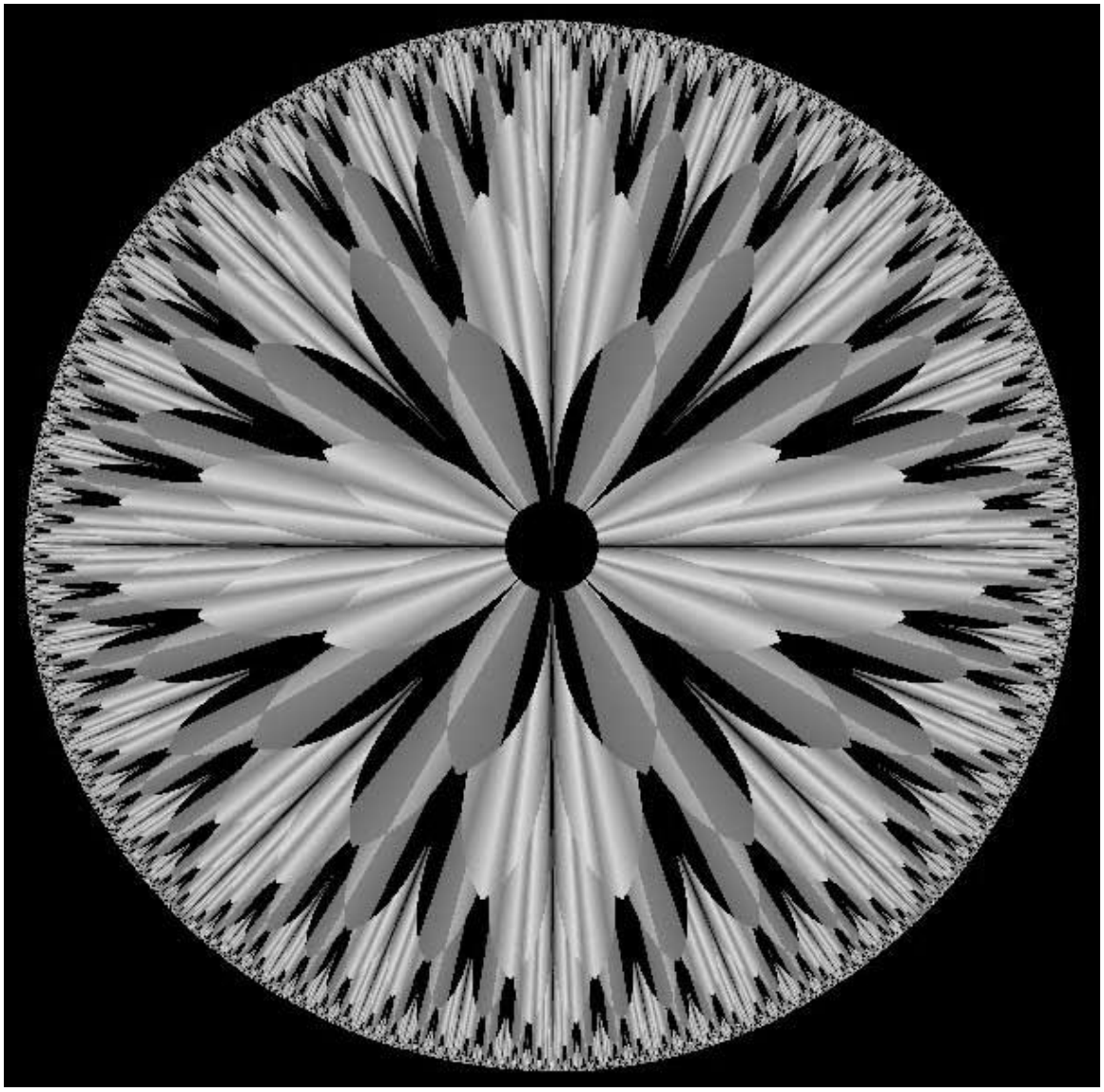
Practical Set-5

Tool: Matlab

1. Apply thresholding and contrast starching on the following image. (File: Original image & Transform images, conclusion).



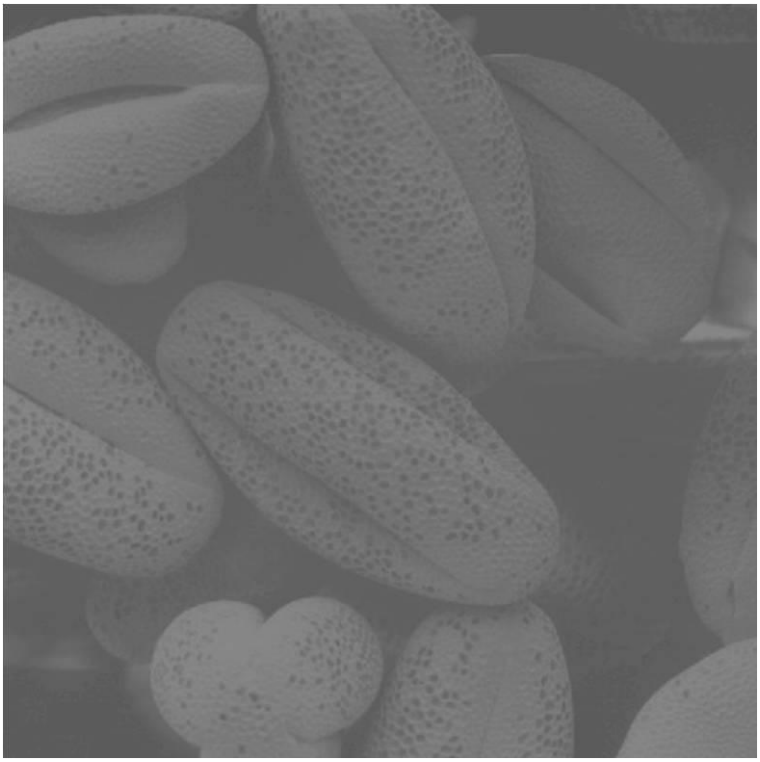
2. Display eight different bit planes of the following image.



3. Apply Histogram Equalization on the image given in (1).

Solution:

1. Apply thresholding and contrast starching on the following image. (File: Original image & Transform images, conclusion).



```
function parc5(a,option)
% Contrast streching & Thresholding
% Read a gray scale image and apply the Contrast streching &
% Thresholding and Give the conclusion.

% Function : Show Original image
% Description : This function is use for the show original image and read
% the image in matrix form.Find the size of the image.check the image rgb
% image or gray scale image.If image is rgb image then convert image into
% grayscale image.
imshow(a);
title('Original image');
img1=imread(a);
[x y z]=size(img1);
if z == 3
   b=rgb2gray(img1);
else
   b=img1;
```

```
end

%Function : Apply contrast streching and Thresholding
%Description : This function is use for apply thresholding and contrast
%streching on image.Option 1 is use for contrast streching and option 2 is
%use for thresholding.
E=20;
m=127;

if option==1
   c = 1./(1+(m./(double(b) + eps)).^E);
   figure;
   imshow(c);
   title('Contrast streching Image');
elseif option==2
   [r c]=size(b);
   for i=1:r
      for j=1:c
         if b(i,j)<=127
            b(i,j) = 0;
         else
            b(i,j)=255;
         end
      end
   end
   figure;
   imshow(b);
   title('Thresholding Image');
else
   display('Give proper Input');
end

end
```

Output:

```
>> parc5one('img7.jpg',1)
```

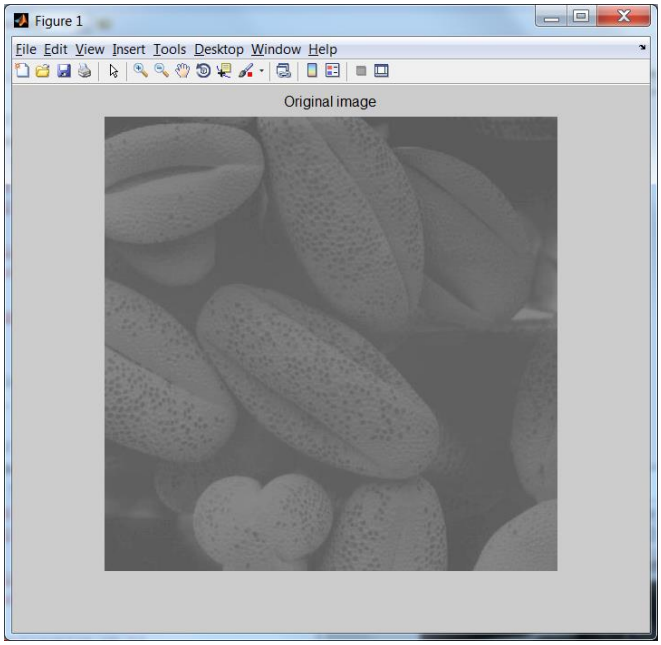


Fig 5:1 Display the original image

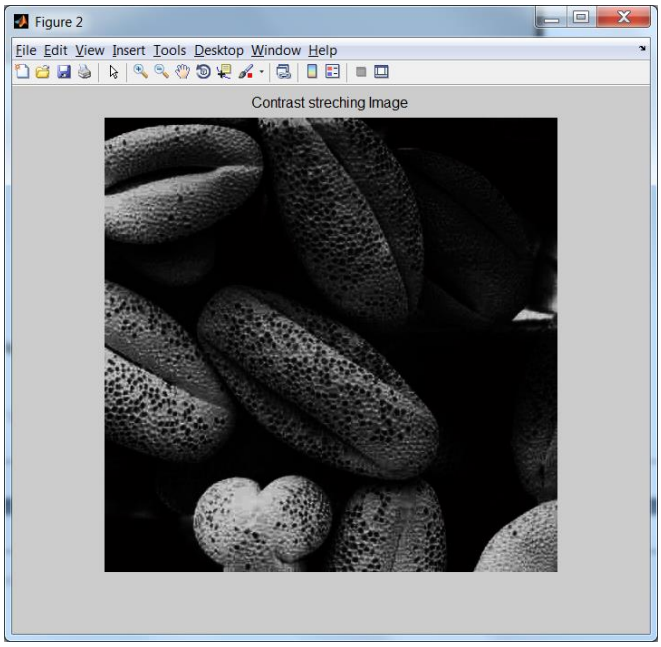


Fig 5:2 display the contrast stretching image

>> parc5one('img7.jpg',2)

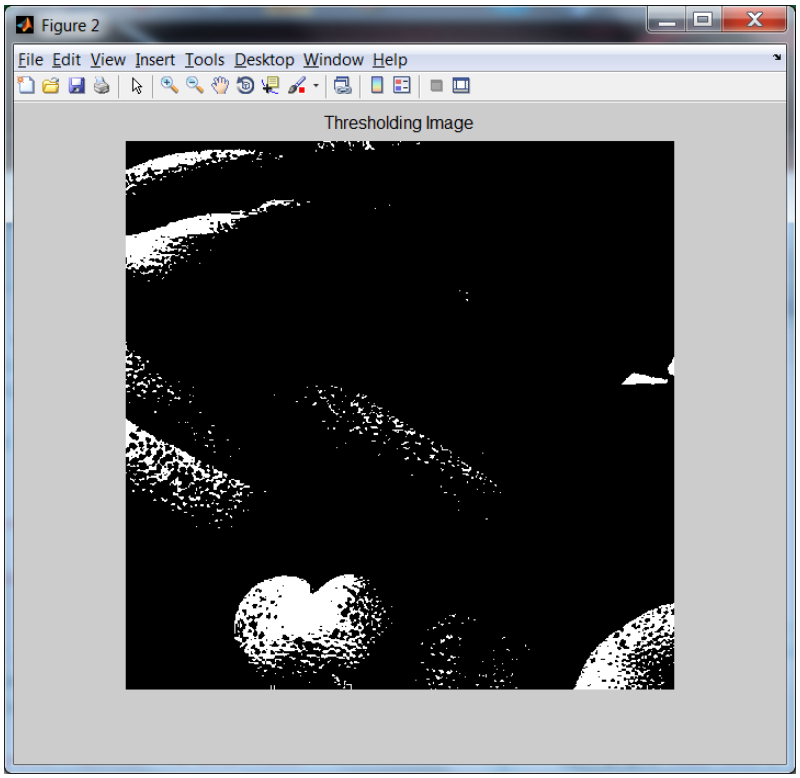
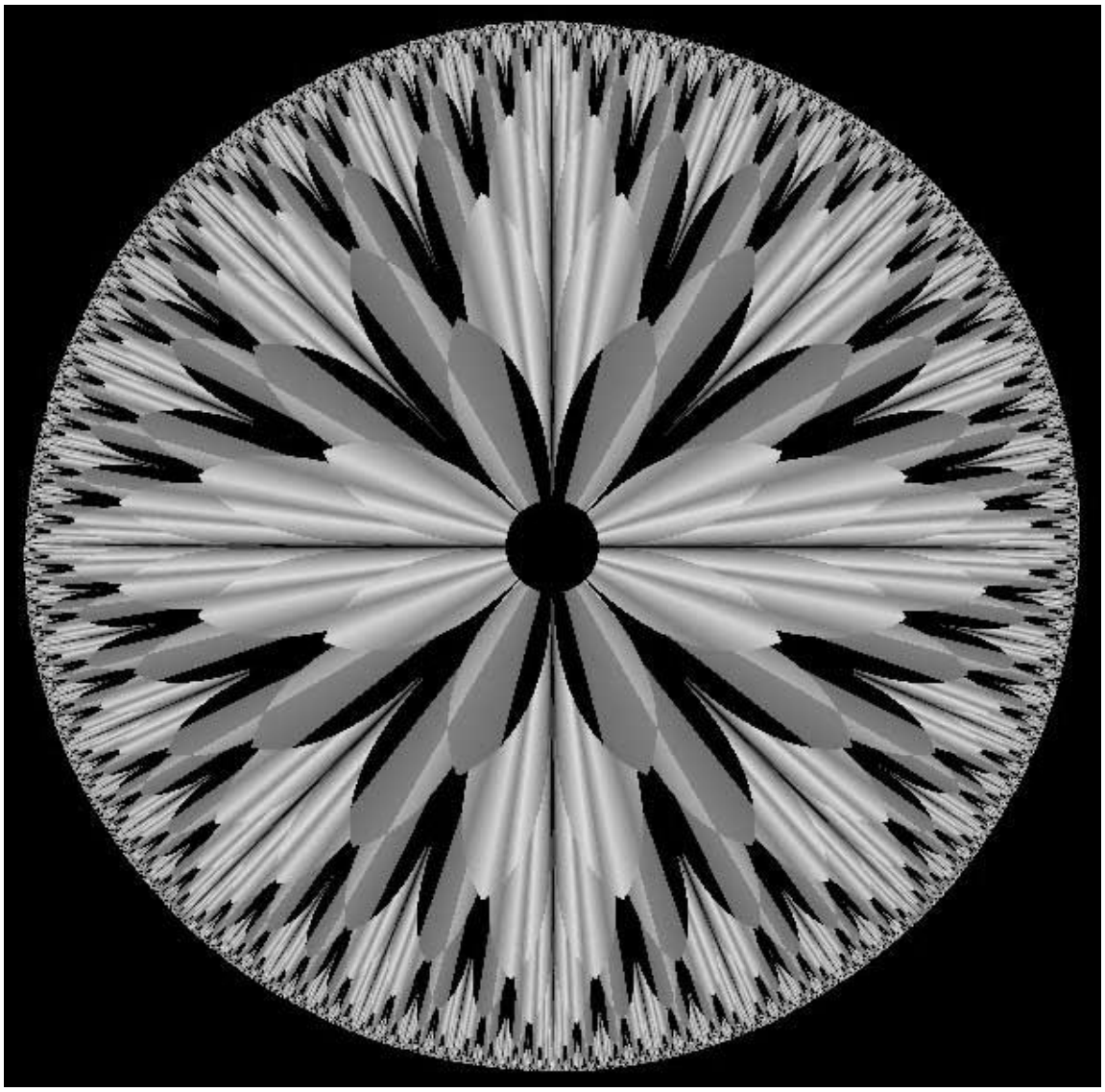


Fig 5:3 display the thresholding image

2. Display eight different bit planes of the following image.



Program:

```
function parc5two(a)
% Bit planes Image
% Display eight different bit planes of the following image
% and Give the conclusion.

% Function : Show Bit planes image
% Description : This function is use for the show bit planes image.If the
% image is rgb image then convert image into grayscale image.
imshow(a),title('original image');
imag1=imread(a);
```

```
img=rgb2gray(imag1);
[x y z]=size(img);
if z==3
   img1=rgb2gray(img);
else
   img1=img;
end

for i=1:8
   figure;
   b=bitget(img1,i);
   imshow(double(b)),title('bit planes image');
end

end
```

Output:

```
>> parc5two('img8.jpg')
```

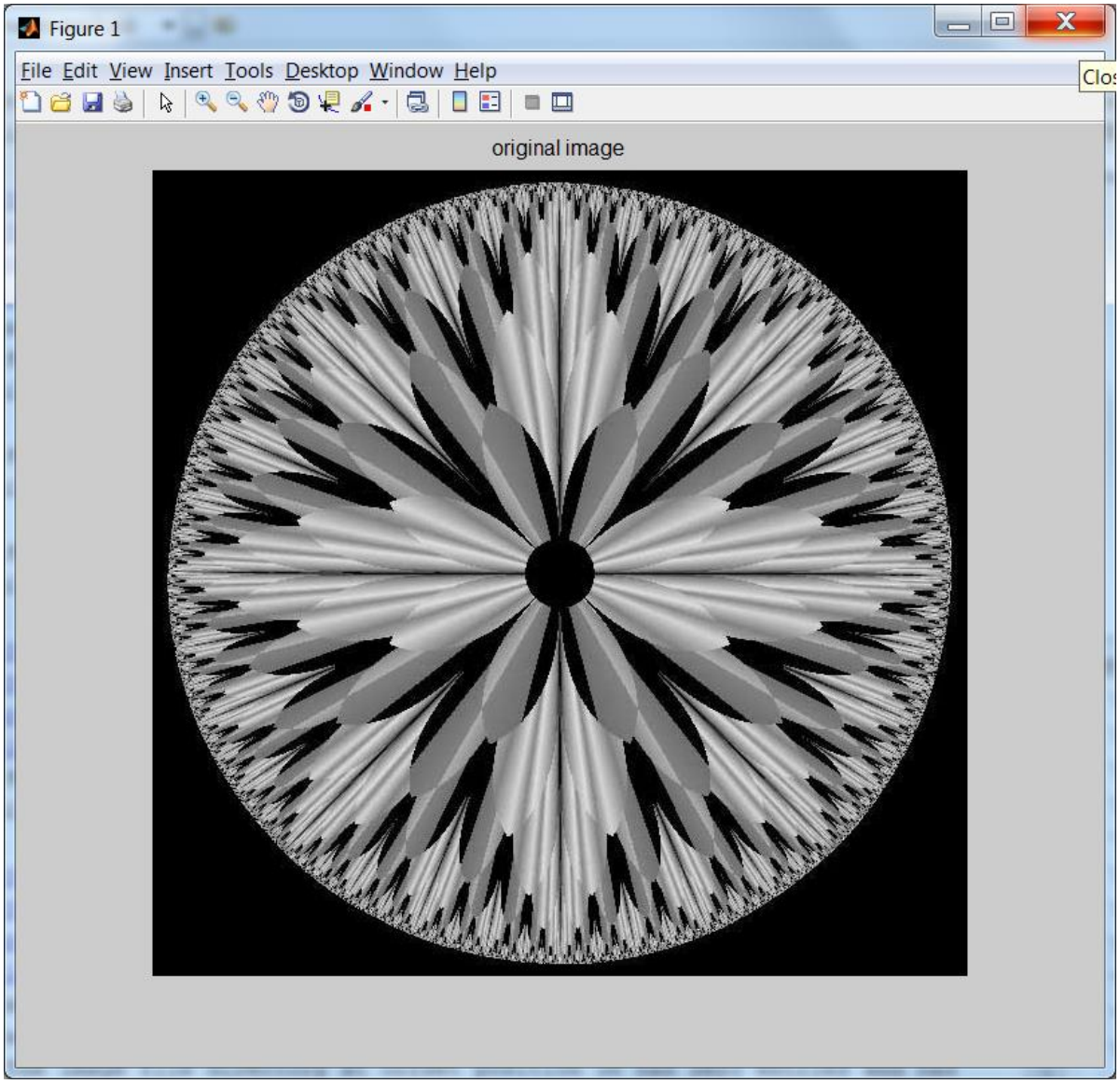


Fig 5.4 Display original image

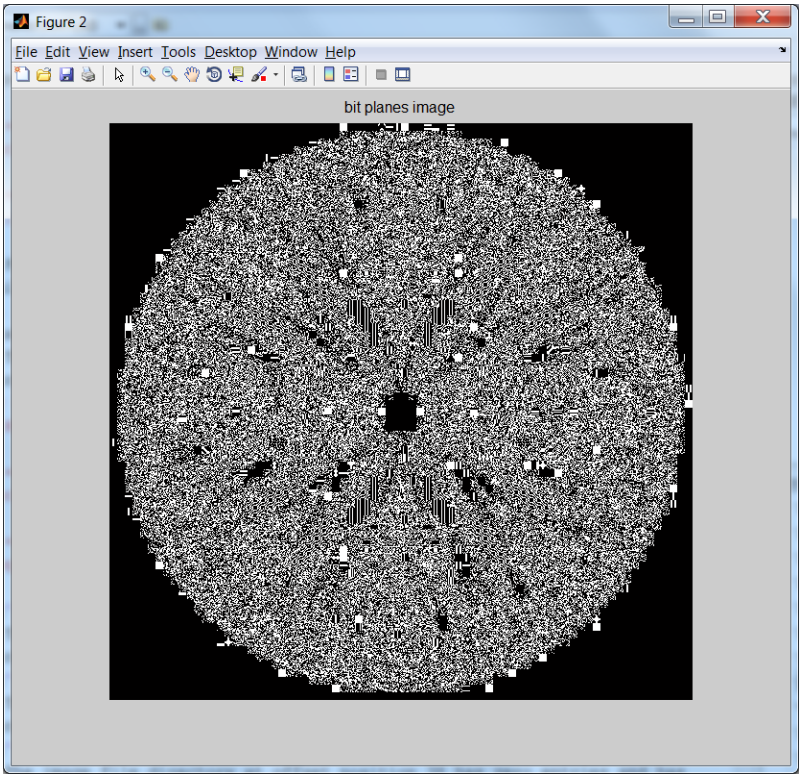


Fig 5:5 display bit planes 1 image

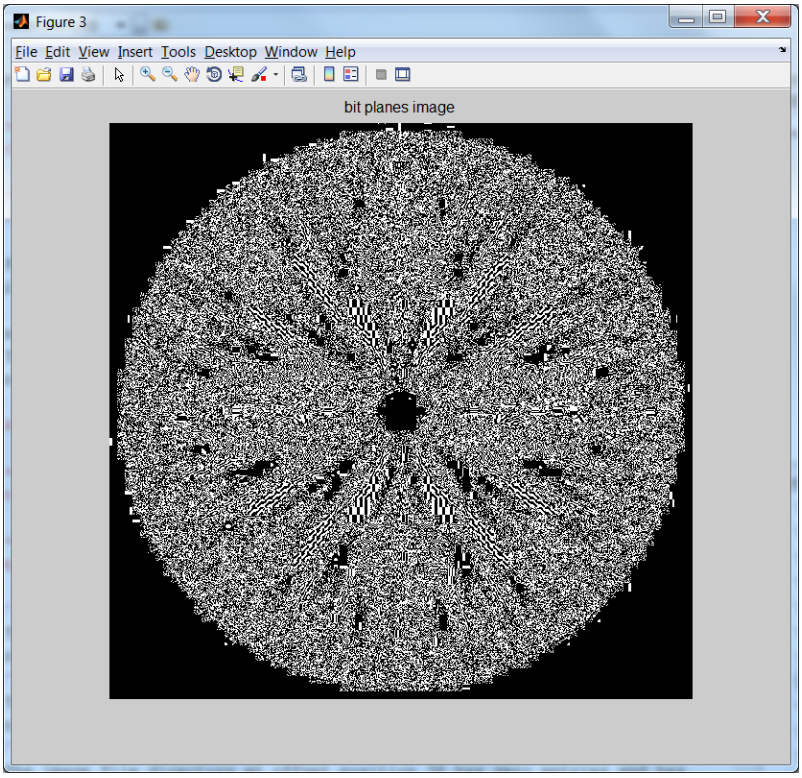


Fig 5:6 display bit planes 2 image

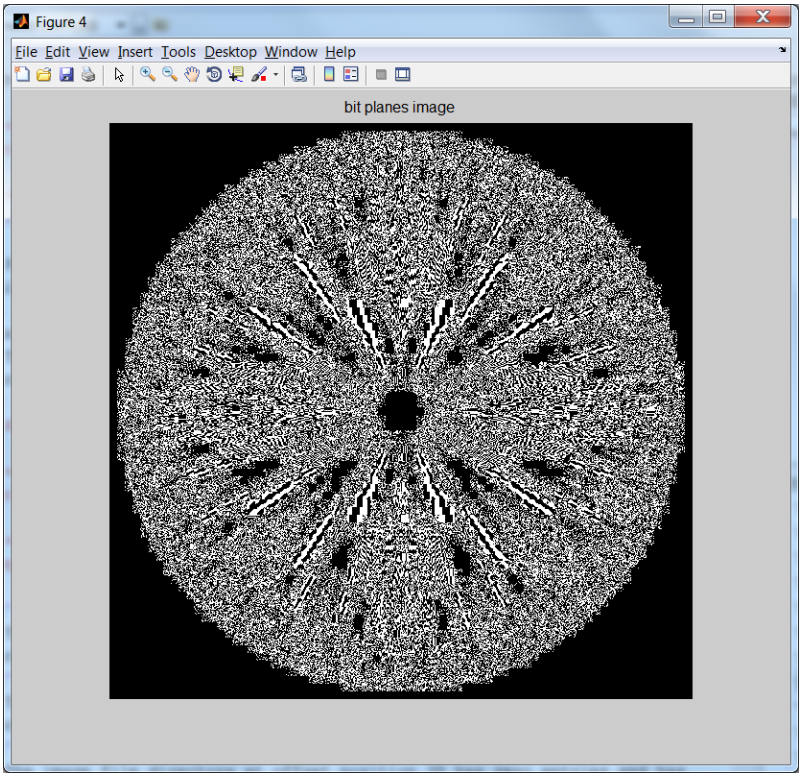


Fig 5:7 display bit planes 3 image

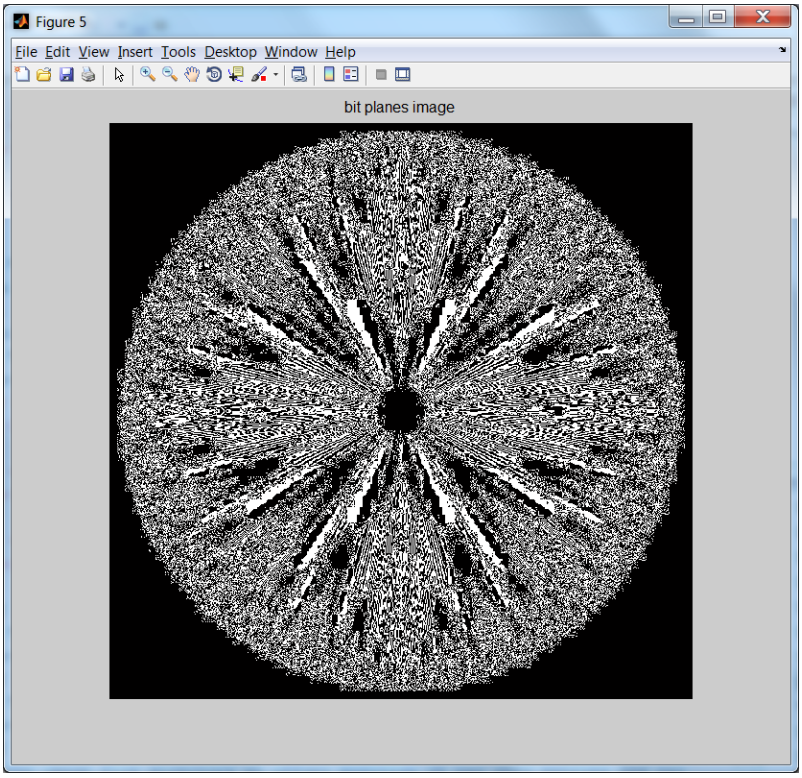


Fig 5:8 display bit planes 4 image

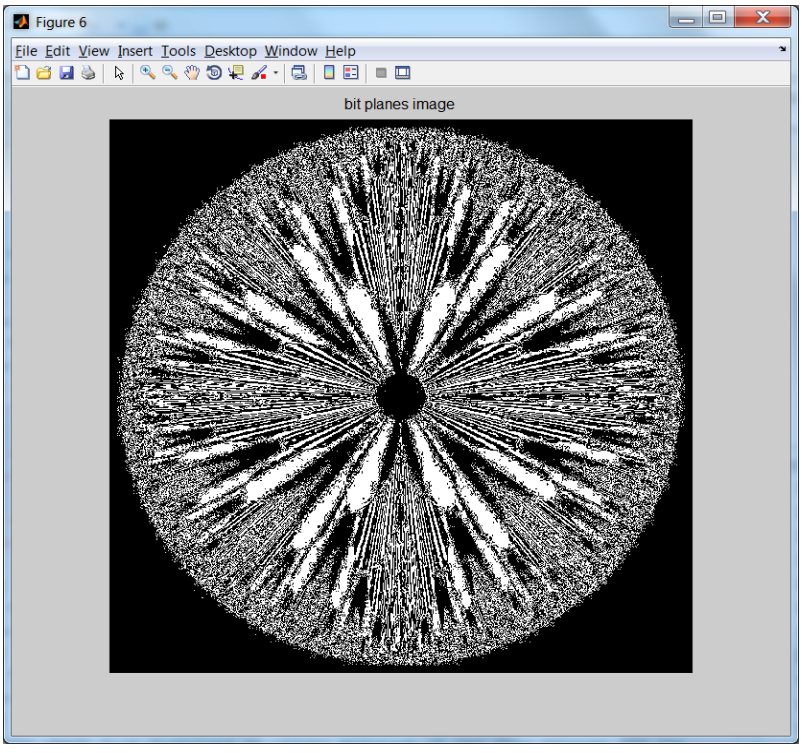


Fig 5:9 display bit planes 5 image

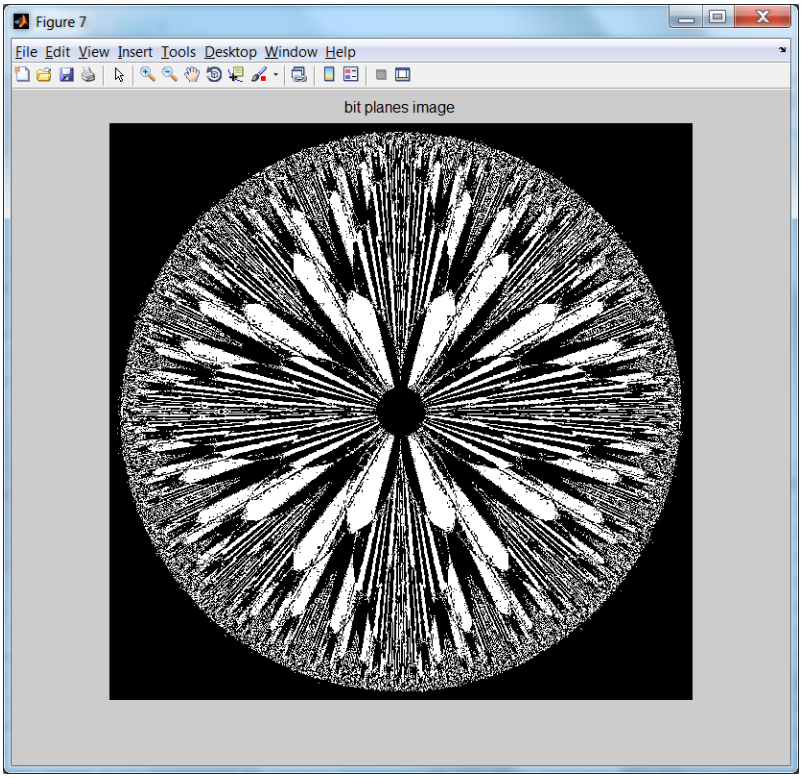


Fig 5:10 display bit planes 6 image

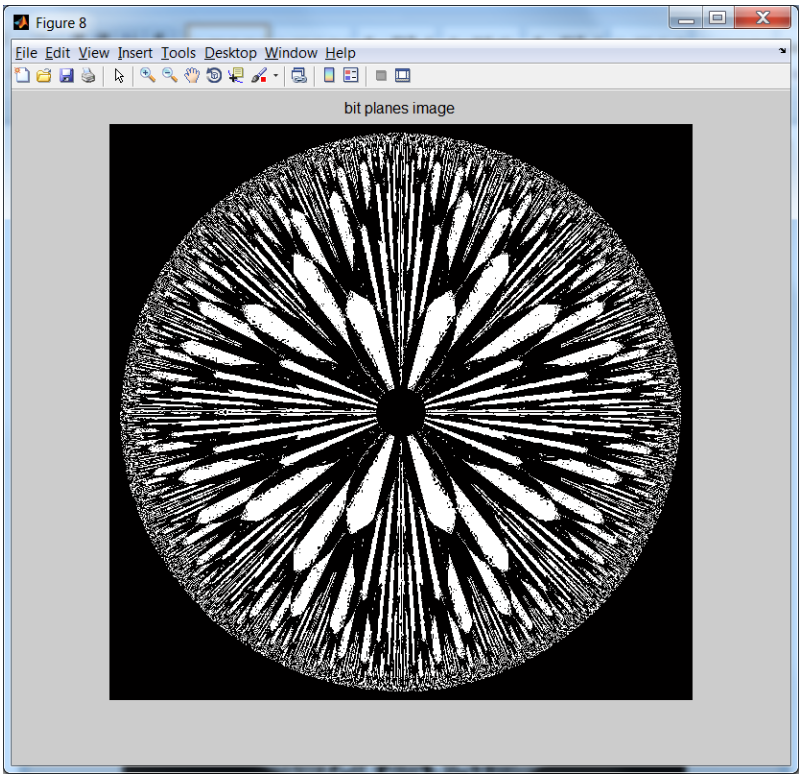


Fig 5:11 display bit planes 7 image

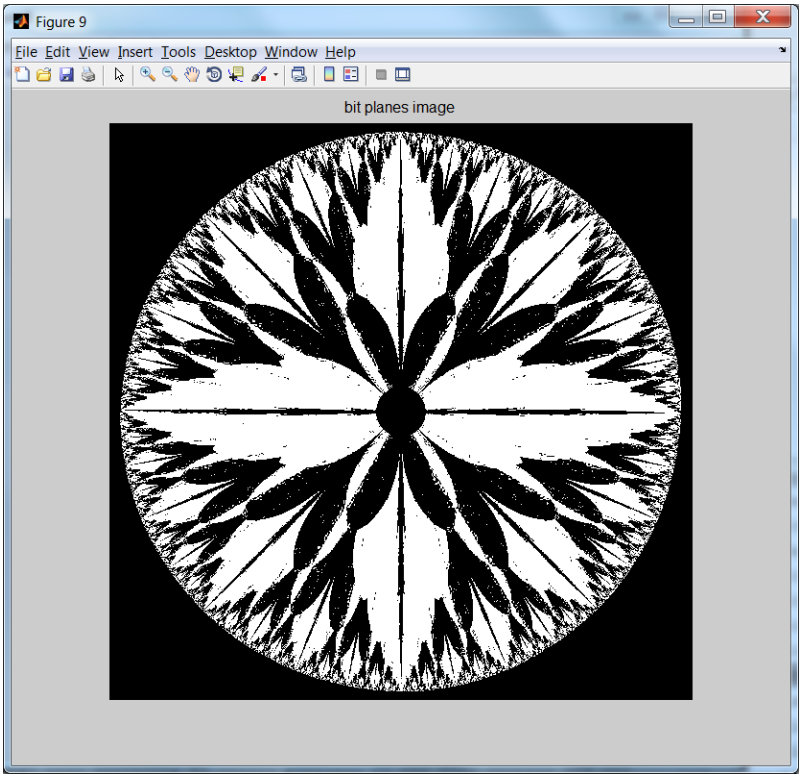


Fig 5:12 display bit planes 8 image

3. Apply Histogram Equalization on the image given in (1).

Program:

```
function equlization(f)
% Histogram Equalization
% Apply Histogram Equalization on the image
% and Give the conclusion.
%
% Function : Show Histogram Equalization image
% Description : This function is use for the Show Histogram Equalization
% image.
imshow(f),title('Original image');
b=imread(f);
a=histeq(b);
figure;
imshow(a);
title('Histogram equalization image');
end
```

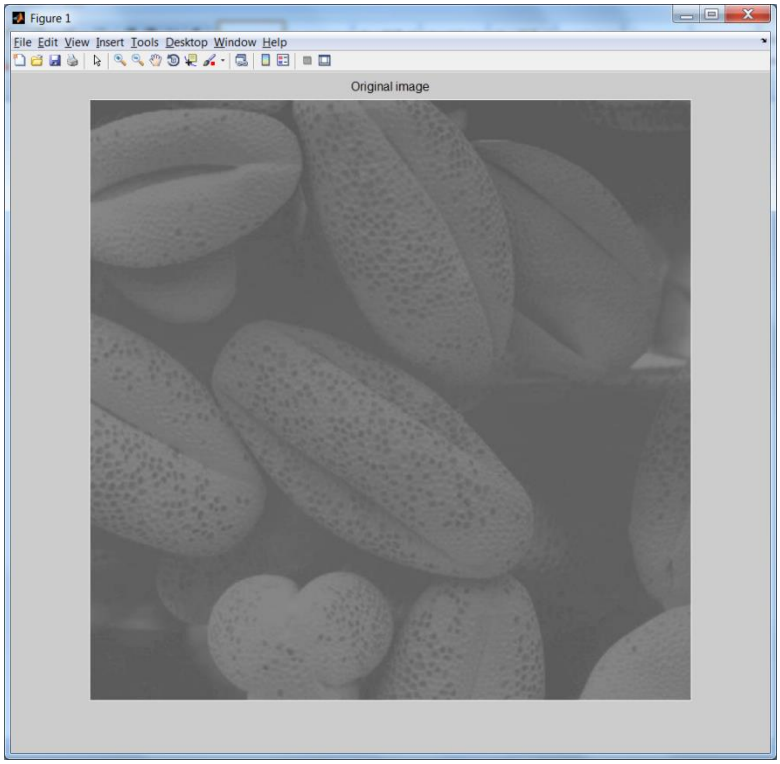


Fig 5:12 display original image

output: >> parc5three('img777.jpg')

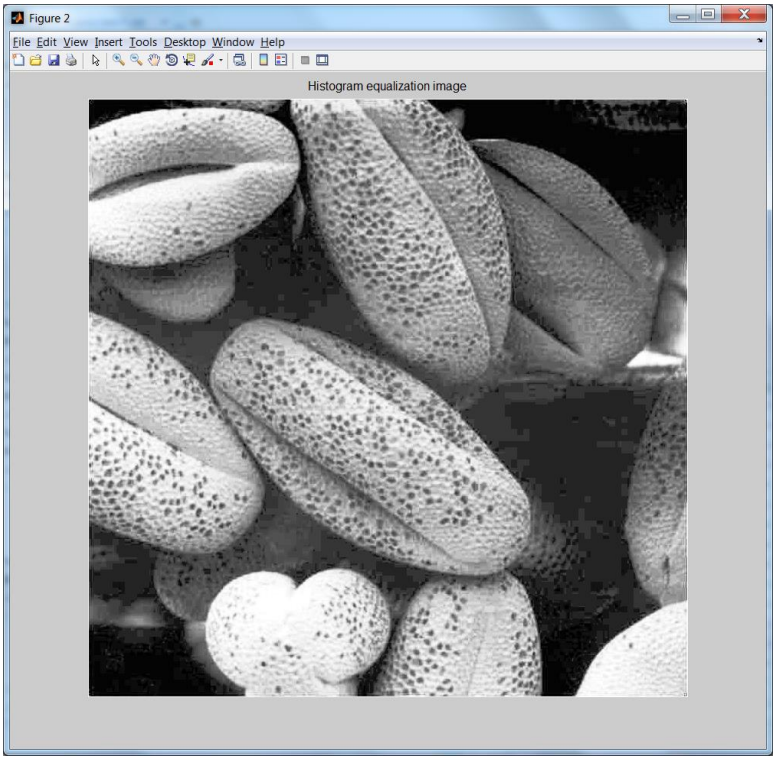
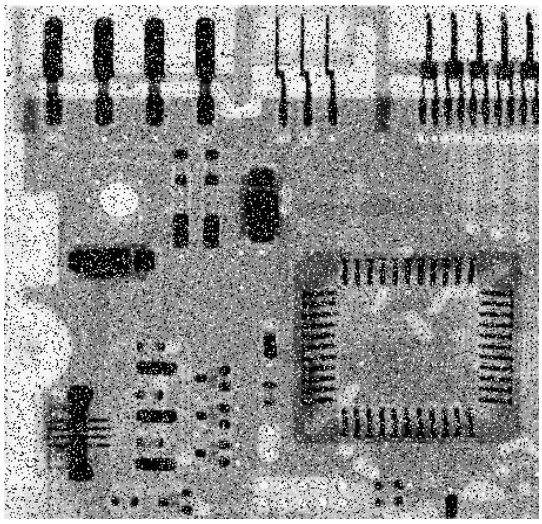


Fig 5:13 display histogram equalization image

Practical Set-6

Tool: Matlab

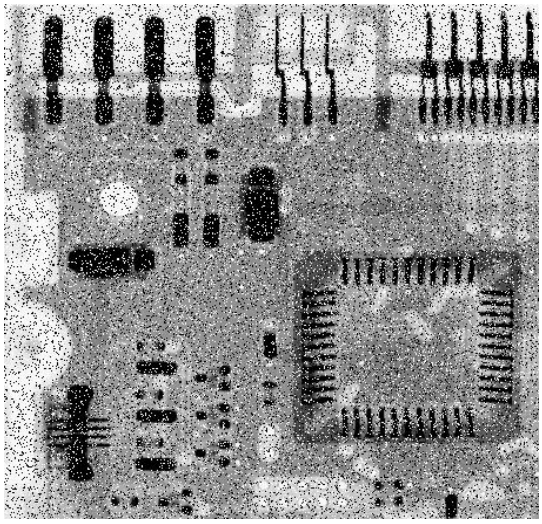
1.) **Apply averaging and median filter of size 3 x 3 on the following image to reduce noise. Compare results.**



2)Apply laplacian filter on the following image. Display laplacian, scale laplacian and sharpen image. Use two possible laplacian filter with center –4 and –8. Compare results.

Solution:

1.) Apply averaging and median filter of size 3 x 3 on the following image to reduce noise. Compare results.



Program:

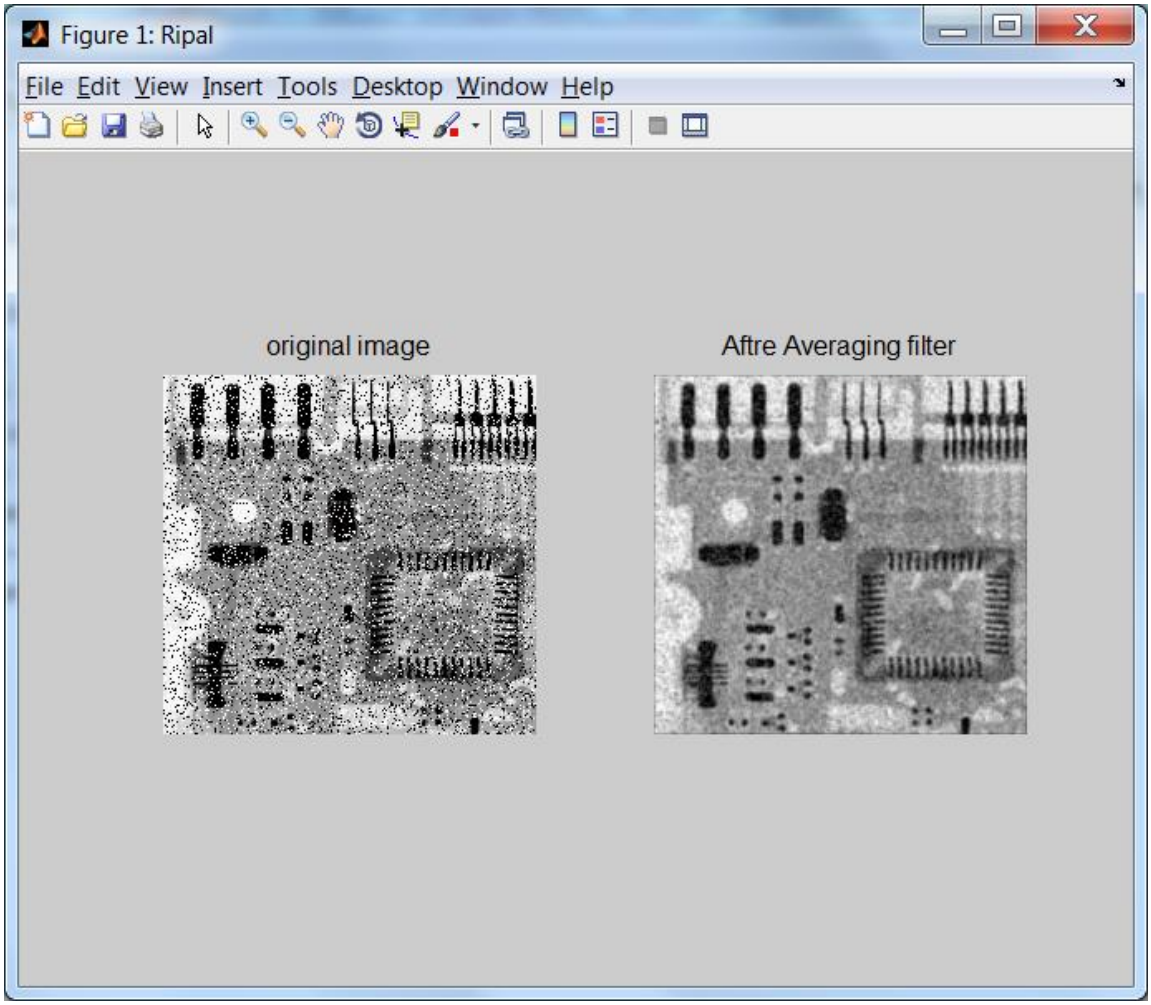
```
function parc6one1()

% Apply averaging and median filter of size 3 x 3 on the following image
% to reduce noise. Compare results.

clc
clear all
disp('This program shows the filtering method')
disp('= = = = = = = = = = = = = = = = = = = = ')
disp('1.averaging filter')
disp('2.median filter')
disp('= = = = = = = = = = = = = = = = = = = = ')

disp('');
m=input('Input the size of the mask:');
disp('');
img1=input('Input the image name:');

[x y z]=size(img1);
if z == 3
   img=rgb2gray(img1);
else
   img=img1;
```

```
end
a=imread(img);

   disp('');
   s=input('Input the choice of the filtering:');

   switch s
      case 1
         ave(m,a);

      case 2
         med(m,a);

      otherwise
         disp('give correct option');
   end
end

% Function:Averaging(avg) Filter
% Description:This function is use for apply filtering on the image using
% the averaging filter.
function ave(m,a)
figure('name','Ripal');
subplot(1,2,1)
imshow(a),title('original image');
d=im2double(a);
h=((inv(power(m,2)))*ones(m,m));
b=imfilter(d,h);
subplot(1,2,2)
imshow(b),title('Aftre Averaging filter');
disp('----------------------------------------------------------------');
disp('The output image give a smoothing than the original image');
end

% Function:Median(med) Filter
% Description:This function is use for apply filtering on the image using
% the median filter.
function med(m,a)
figure('name','Ripal');
subplot(1,2,1)
imshow(a),title('original image');
d=im2double(a);
B=medfilt2(d,[m,m]);
subplot(1,2,2)
imshow(B),title('After median filter');
disp('----------------------------------------------------------------');
disp('The output image give a smoothing than the original image');
disp('Median filter is much better suitesd than avering for the removel');
disp('of salt and pepper noise');
```

```
end
```

Output:

```
>>parc6one1

This program shows the filtering method

= = = = = = = = = = = = = = = = = = = =

1.averaging filter

2.median filter

= = = = = = = = = = = = = = = = = = = =

Input the size of the mask:5

Input the image name:'parc6img.jpg'

Input the choice of the filtering:1

------------------------------------------------------------------

The output image give a smoothing than the original image
```



>>parc6one1

This program shows the filtering method

= =

1.averaging filter

2.median filter

= =

Input the size of the mask:5

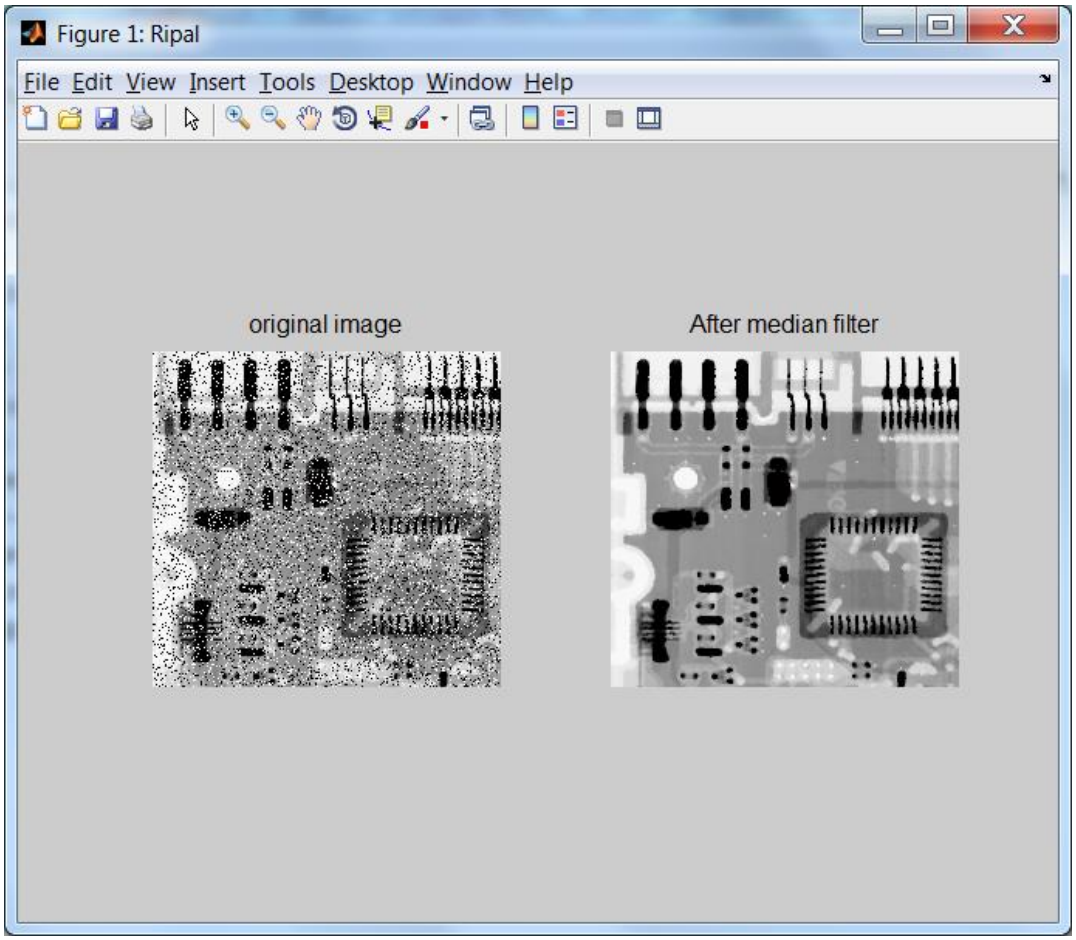
Input the image name:'parc6img.jpg'

Input the choice of the filtering:2

The output image give a smoothing than the original image

Median filter is much better suitesd than avering for the removel

of salt and pepper noise



2)Apply laplacian filter on the following image. Display laplacian, scale laplacian and sharpen image. Use two possible laplacian filter with center –4 and –8. Compare results.

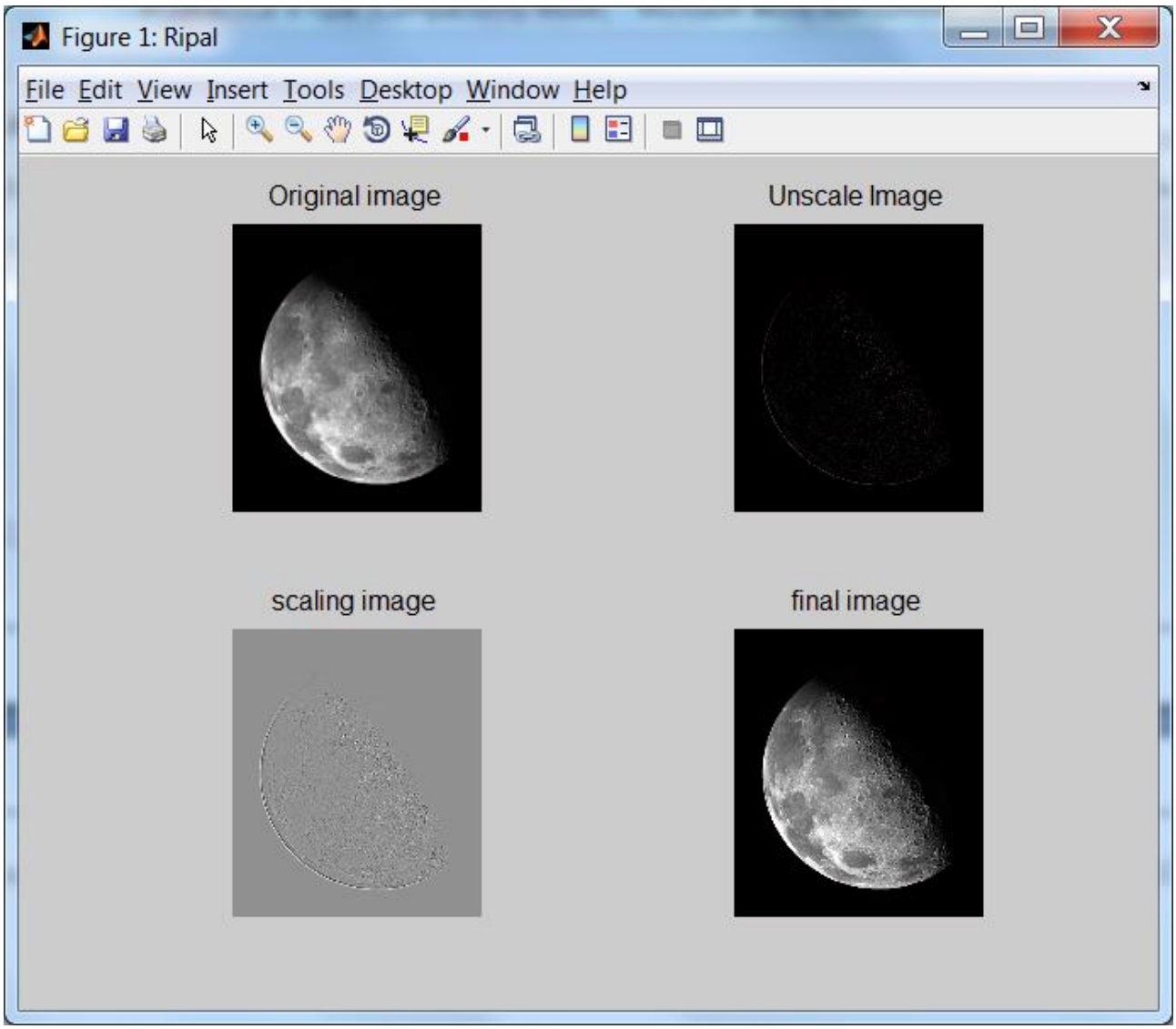
Program:

```
function parc6two()

% Function:Laplcian filter
% Description:This function is use for the apply laplcian filter with
% center -4 and -8 on the image.First show the slightly blurred original
% image.second figure shows that the result of filtering this image with
% laplacian mask  or shows unscale image.third figure shows scaling image.
% fourth figure shows that result of adding scaling image and original
% image.

clc
clear all
disp('This program use for Apply laplacian filter on the image')
disp('= = = = = = = = = = = = = = = = = = = = = ')
disp('1.laplacian filter with center –4')
disp('2 laplacian filter with center –8')
disp('= = = = = = = = = = = = = = = = = = = = = = = ')

disp('');
img1=input('Input the image name:');


[x y z]=size(img1);
if z == 3
   img=rgb2gray(img1);
else
   img=img1;
end
a=imread(img);
disp('');
s=input('Input the choice of the filtering:');

   switch s
      case 1
```

```
        h=[0 1 0;1 -4 1;0 1 0];
        lap_filt(h,a);
        disp('-----------------------------------------------------');
        disp('The detail in final image is unmistakably clearer and');
        disp('sharper than original image');

    case 2
        h=[1 1 1;1 -8 1;1 1 1];
        lap_filt(h,a);
        disp('-----------------------------------------------------');
        disp('The detail in final image is unmistakably clearer and');
        disp('sharper than original image,here note the significant');
        disp('improvement in sharpness compare the laplacian')
        disp('filter with center -4');

    otherwise
        disp('give the correct option for filter')
  end
end

function lap_filt(h,a)
figure('name','Ripal');
subplot(2,2,1);
imshow(a),title('Original image');

b=imfilter(double(a),h);
subplot(2,2,2);
imshow(uint8(b)),title('Unscale Image');

mini=min(min(b));
%sprintf('%d',mini)
img1=b-mini;
maxi=max(max(img1));
%sprintf('%d',maxi)
img2=(255./maxi).*img1;
subplot(2,2,3);
imshow(uint8(img2)),title('scaling image');

final=double(a)-b;
subplot(2,2,4)
imshow(uint8(final)),title('final image');

end
```

Output:

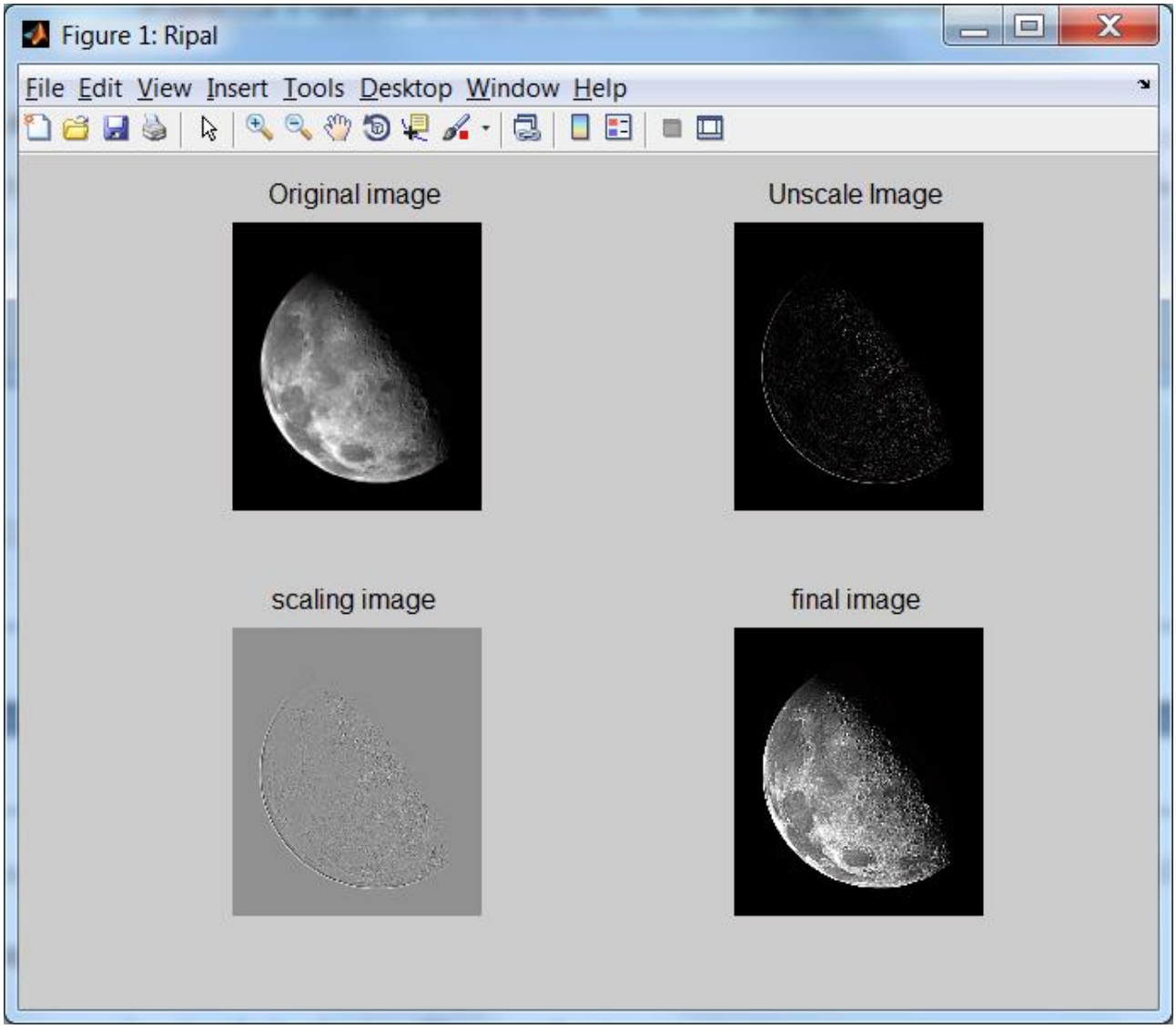
```
>> parc6two

This program use for Apply laplacian filter on the image
= = = = = = = = = = = = = = = = = = = = =
1.laplacian filter with center –4
2 laplacian filter with center –8
= = = = = = = = = = = = = = = = = = = = =
Input the image name:'moon.jpg'
Input the choice of the filtering:1
-----------------------------------------------------
The detail in final image is unmistakably clearer and
sharper than original image
```



>>parc6two

This program use for Apply laplacian filter on the image
= =
1.laplacian filter with center –4
2 laplacian filter with center –8
= =
Input the image name:'moon.jpg'
Input the choice of the filtering:2

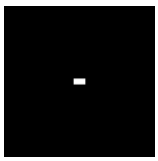
The detail in final image is unmistakably clearer and sharper than original image,here note the significant improvement in sharpness compare the laplacian filter with center -4



Practical Set-7

Tool: Matlab

- Generate Fourier Spectrum, Centered Fourier Spectrum and visually enhanced Centered Fourier Spectrum for the following image. Also display wireframe model of the centered Fourier spectrum Using Inverse Fourier transform, display original image from its Fourier transform.



- Write a user define function genmeshgrid(M,N). If M=N=3, output should be,

  2 1 2

  1 0 1

  2 1 2

- Write a user define function lpfilter(type,M,N,D_0,n). Here, "type" is the type of low pass filter, M and N are the number of rows and columns in the filter, D_0 is the cut off frequency and n specifies the order of the filter.

- Write a user define function hpfilter(type,M,N,D_0,n). Here, "type" is the type of low pass filter, M and N are the number of rows and columns in the filter, D_0 is the cut off frequency and n specifies the order of the filter.

- Apply gaussian low pass filter with and without padding on the following image. Compare results. Cut of Frequency should be 10 when we are applying it without padding.

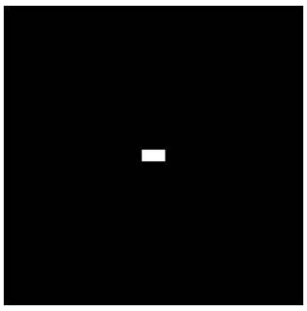
- Apply different low pass and high pass filters on the following image with radius 5, 20, 35, 75, 225. Compare results.

- Generate sobel filter of size 1200 x 1200 in frequency domain corresponding to the vertical sobel filter of size 3 x 3 in the spatial domain. Display centered and non-centered fourier spectrum of the filter (In terms of wireframe plot and normal plot).

- Apply vertical sobel filter (in spatial domain) of size 3 x 3 on the following image. Also apply corresponding frequency domain filter. Compare results.

Solution:

1. **Generate Fourier Spectrum, Centered Fourier Spectrum and visually enhanced Centered Fourier Spectrum for the following image. Also display wireframe model of the centered Fourier spectrum Using Inverse Fourier transform, display original image from its Fourier transform.**

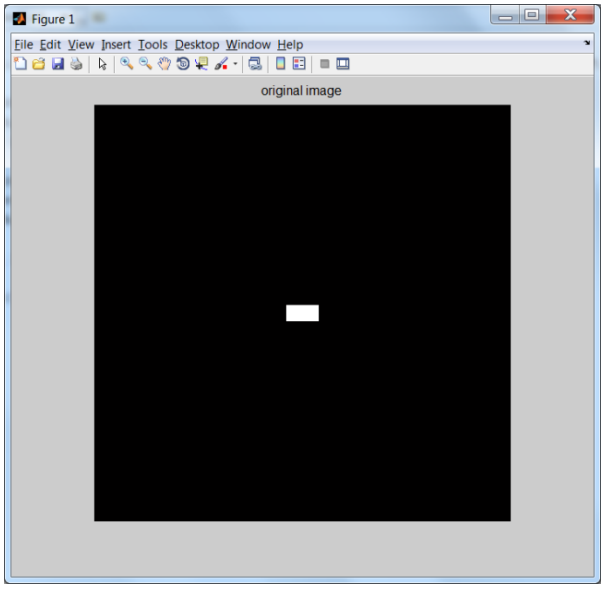


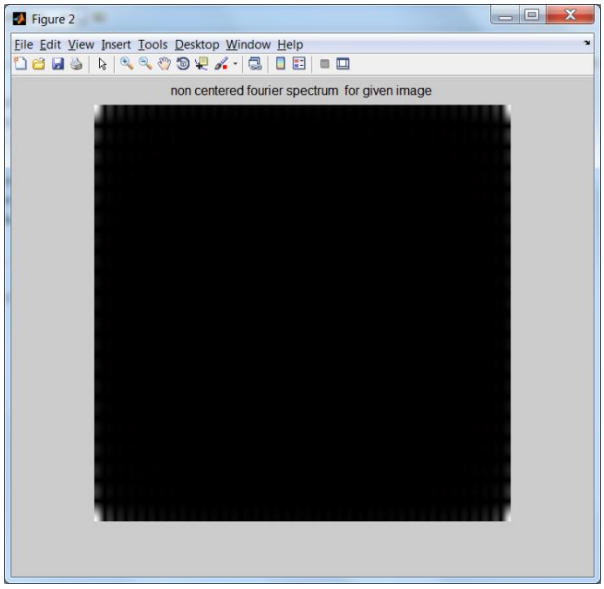
Program:

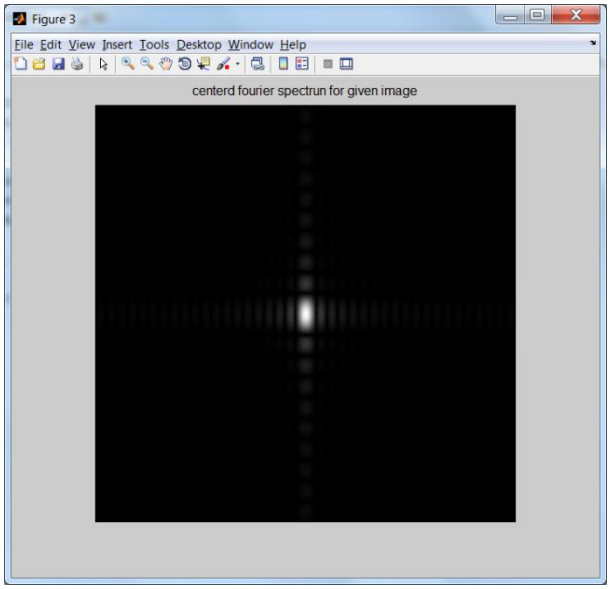
```
function parc7one(image)
imshow(image);
title('original image');
img=imread(image);
%implement fourier transt=formation
ft=fft2(img);

%ft contaion non centered fourier transformed image
%generate fourier transform of non centerd fourier transformed image
fs=abs(ft);%we are having absolute values in ft
figure();
mshow(fs,[]);
title('non centered fourier spectrum  for given image');

%generate centered fourier spectrum
%fftshift will convert centered image to non centered and vice versa
cfs=fftshift(fs);
figure();
imshow(cfs,[]);
title('centerd fourier spectrun for given image');

%visually enhance fourier spectrum
s=im2uint8(mat2gray(log(1+double(cfs))));
```

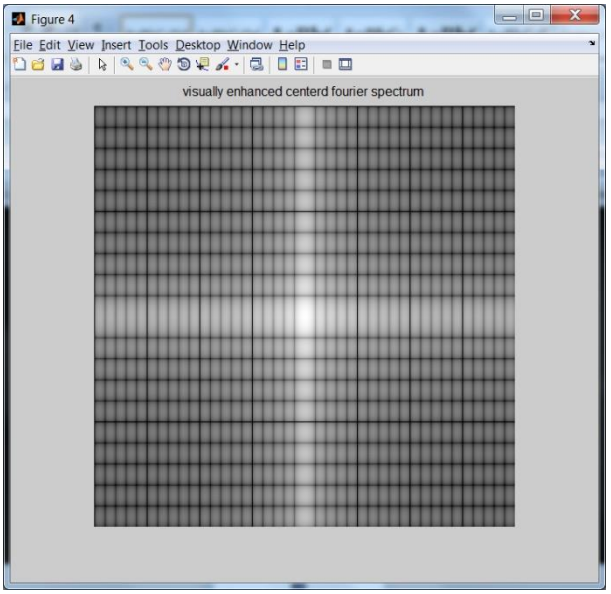
```
figure();
imshow(s,[]);
title('visually enhanced centerd fourier spectrum');

%wireframe model of centerd fourier spectrum
[r c]=size(cfs);
figure();
mesh(cfs(1:20:r,1:20:c));
title('wireframe model');

%inverse transform function to generate actual image
a=ifft2(ft);%ft fourier transformed non centerd image
%ifft2 will convert image frequency domain to special domain
figure();
imshow(a,[]);
title('inverse fourier transformed image');
end
```

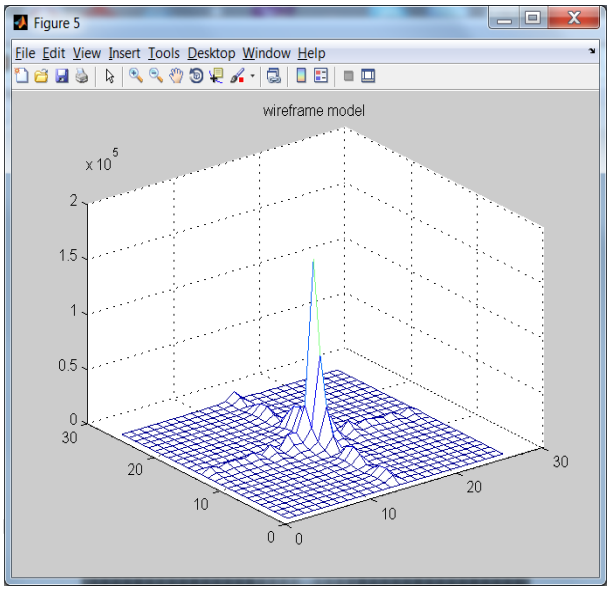
Output >> parc7one('fig1.jpg')

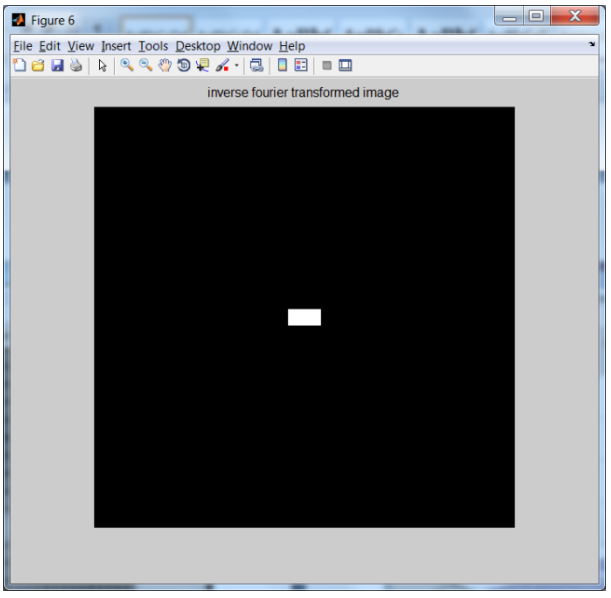












2. Write a user define function genmeshgrid(M,N). If M=N=3, output should be

2 1 2
1 0 1
2 1 2

Program:

```
function D=genmesh(M,N)
cx=uint8(M/2);
cy=uint8(N/2);
for i=1:M
   for j=1:N
      D(i,j)=((double(cx)-i).^2)+((double(cy)-j).^2);
   end
end
end
```

Output:

```
>> genmesh(3,3)
ans =
        2       1       2
        1       0       1
        2       1       2
```

3. Write a user define function lpfilter(type,M,N,D0,n). Here, "type" is the type of low pass filter, M and N are the number of rows and columns in the filter, D0 is the cut off frequency and n specifies the order of the filter.

Program:

```
function [H D]=lpfilter(type,M,N,D0,n)
D=genmesh(M,N);
switch type
   case 'ideal'
      H=double(D<=D0);
   case 'butterworth'
      if nargin == 4
         n=1;
      end
      H=1./(1+(D+D0).^(2*n));
   case 'gaussian'
      H=exp(-(D.^2)./(2*(D0^2)));
   otherwise
      error('not valid filter');
end
end
```

Output:

```
>> lpfilter('ideal',3,3,1,2)
ans =
         0         1         0
         1         1         1
         0         1         0

>> lpfilter('butterworth',3,3,1)
ans =
    0.1000    0.2000    0.1000
    0.2000    0.5000    0.2000
    0.1000    0.2000    0.1000

>> lpfilter('gaussian',3,3,1,2)
ans =
    0.1353    0.6065    0.1353
    0.6065    1.0000    0.6065
    0.1353    0.6065    0.1353
```

4. Write a user define function hpfilter(type,M,N,D0,n). Here, "type" is the type of low pass filter, M and N are the number of rows and columns in the filter, D0 is the cut off frequency and n specifies the order of the filter.

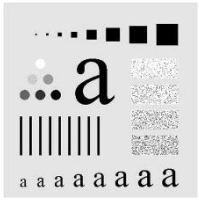
Program:

```
function H=hpfilter(type,M,N,D0,n)
if nargin==4
      n=1;
end
lp=lpfilter(type,M,N,D0,n);
H=1-lp;
End
```

Output:

```
>> hpfilter('ideal',3,3,1,2)
ans =
        1        0        1
        0        0        0
        1        0        1

>> hpfilter('gaussian',3,3,1,2)
ans =
        0.8647    0.3935    0.8647
        0.3935         0    0.3935
        0.8647    0.3935    0.8647

>> hpfilter('butterworth',3,3,1)
ans =
        0.9000    0.8000    0.9000
        0.8000    0.5000    0.8000
        0.9000    0.8000    0.9000
```

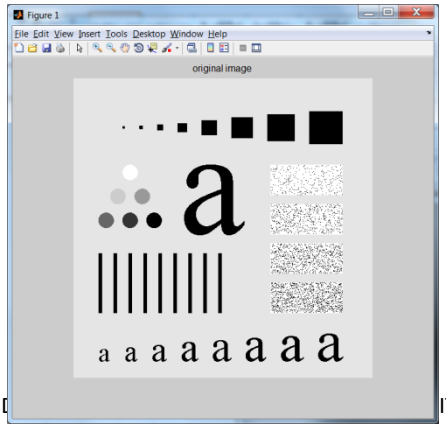
5. Apply different low pass and high pass filters on the following image with radius 5, 20, 35, 75, 225. Compare results.



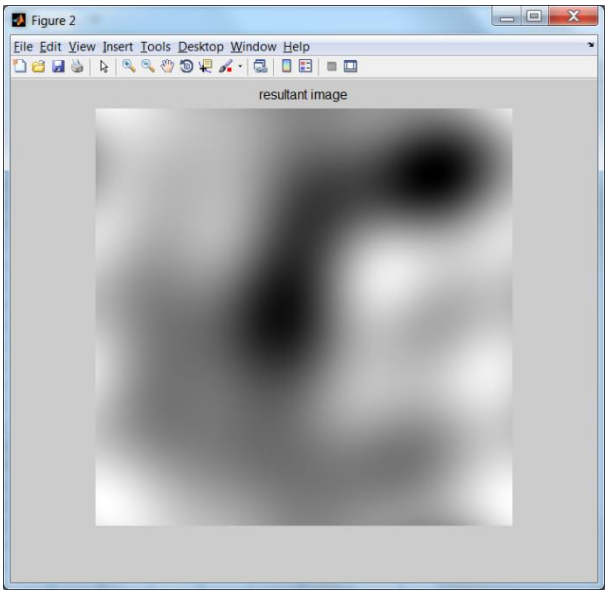
Program:

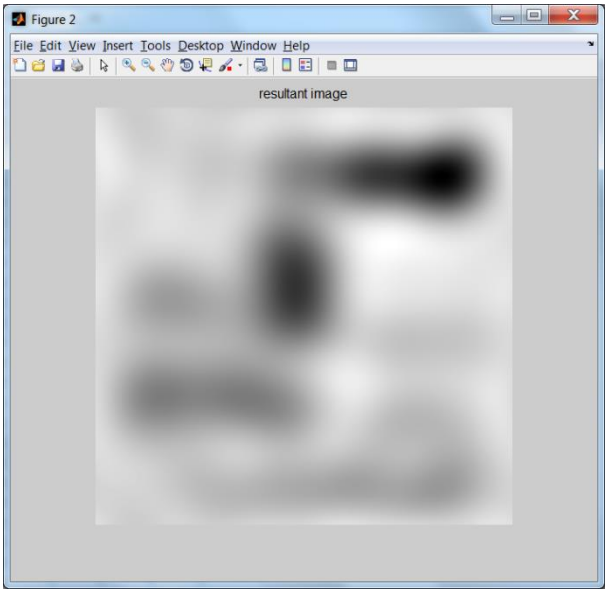
```
function H=prac7_5(image,filter,type,M,N,D0,n)
imshow(image);
title('original image');
img=imread(image);
[r c]=size(img);
M=r; N=c;
if nargin==6
   n=1;
end
if filter == 'lp'
   H=lpfilter(type,M,N,D0,n);
else if filter == 'hp'
   H=hpfilter(type,M,N,D0,n);
end
image1=fft2(img);
CH=fftshift(H);
final=image1.*CH;
a=ifft2(final);
z=real(a);
figure();
imshow(z,[]);
title('resultant image');
end
```

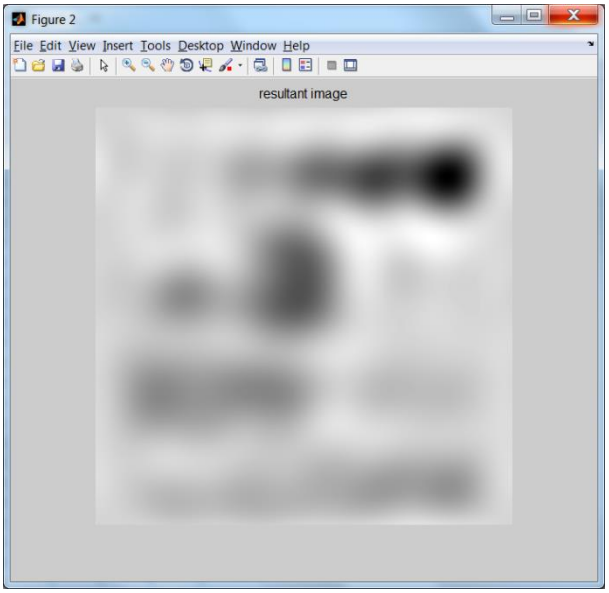
Output:

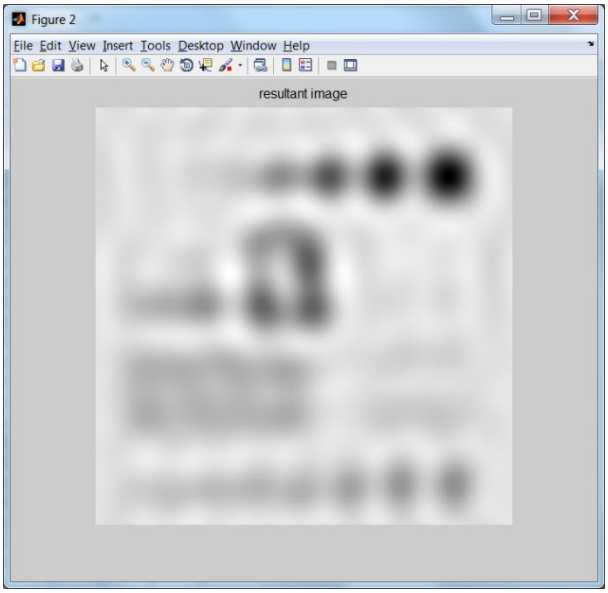


prac7_5('fig4.tif','lp','ideal',3,3,5)
prac7_5('fig4.tif','lp','ideal',3,3,20)
prac7_5('fig4.tif','lp','ideal',3,3,35)
prac7_5('fig4.tif','lp','ideal',3,3,75)
prac7_5('fig4.tif','lp','ideal',3,3,225)

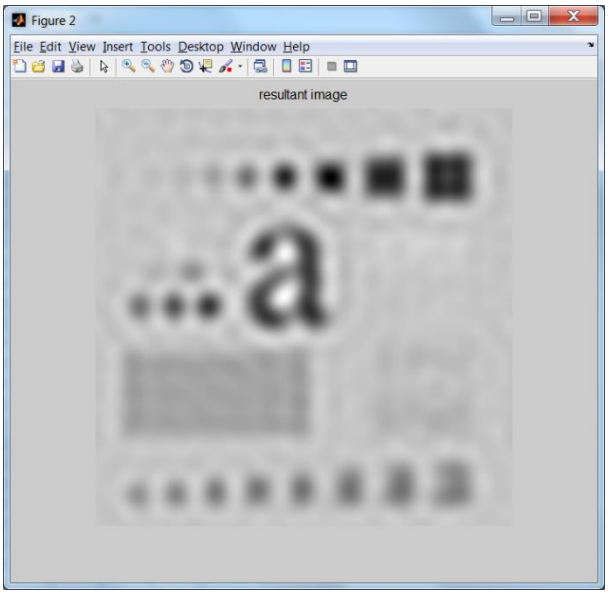




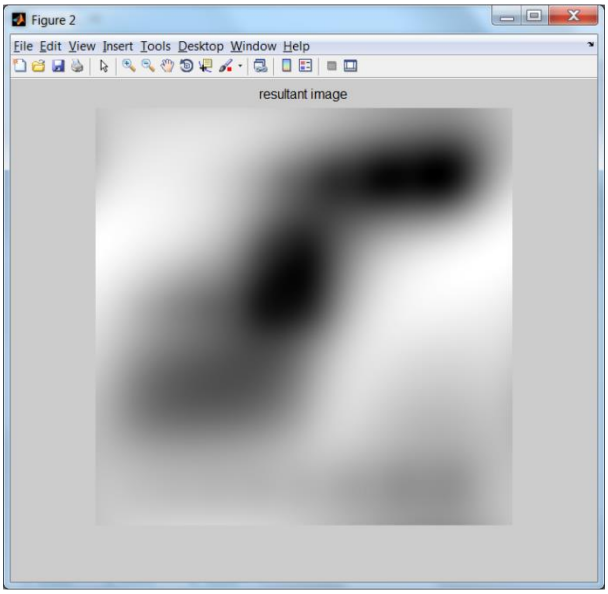


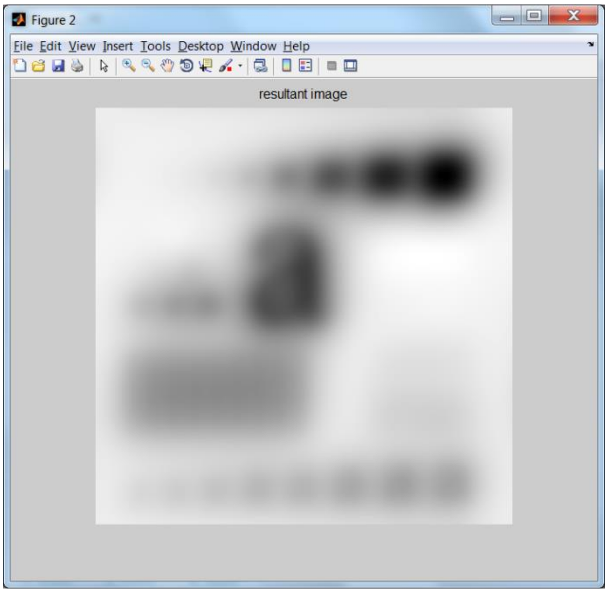


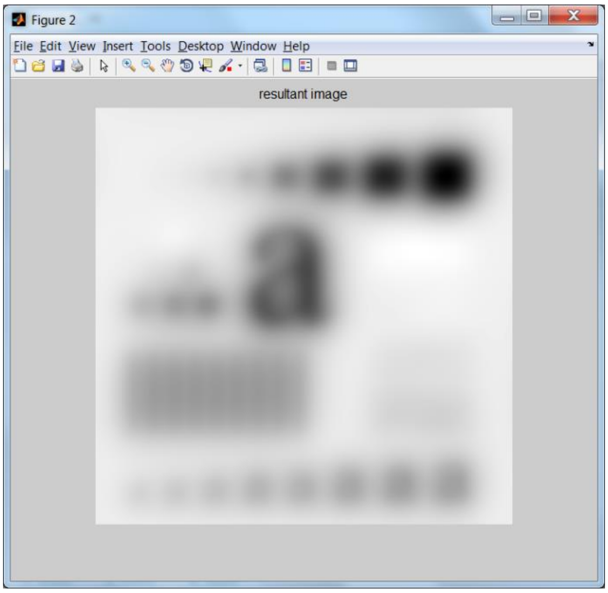
r

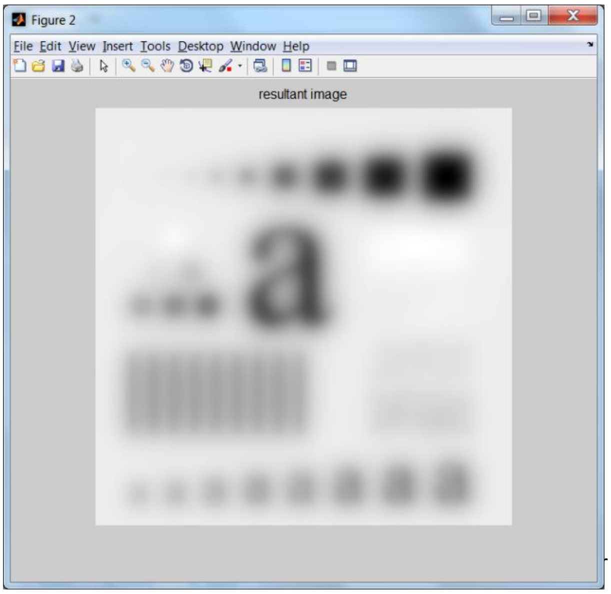


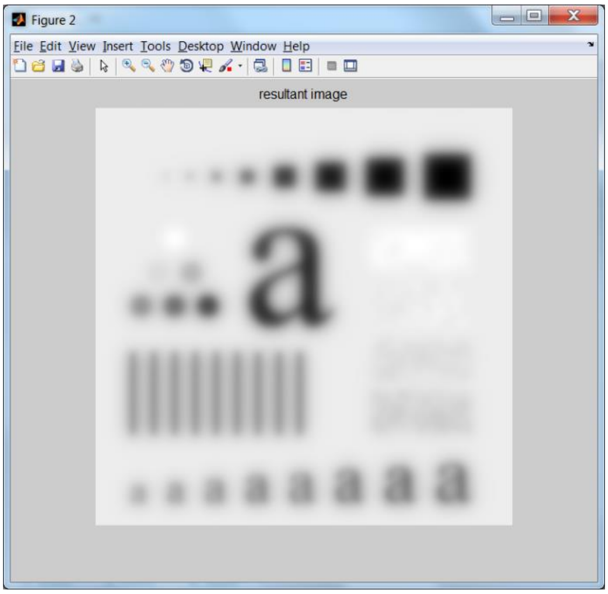
prac7_5('fig4.tif','lp','butterworth',3,3,5)
prac7_5('fig4.tif','lp','butterworth',3,3,20)
prac7_5('fig4.tif','lp','butterworth',3,3,35)
prac7_5('fig4.tif','lp','butterworth',3,3,75)
prac7_5('fig4.tif','lp','butterworth',3,3,225)



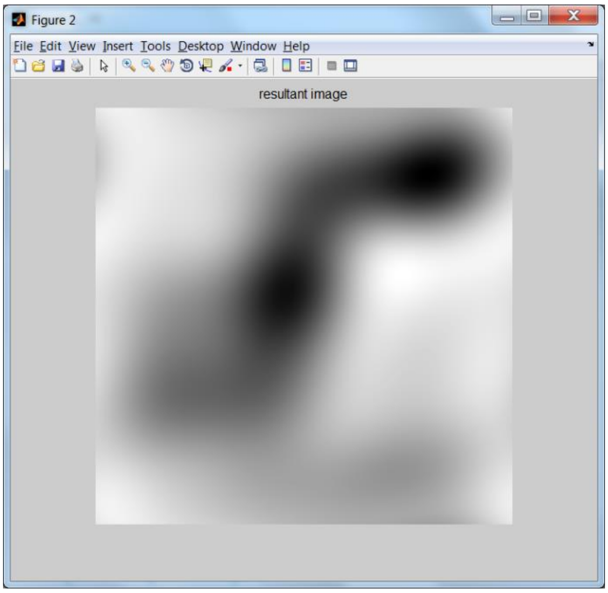


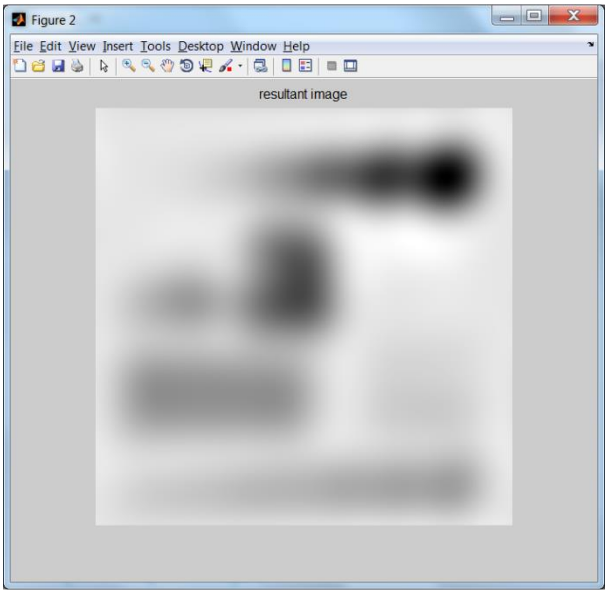


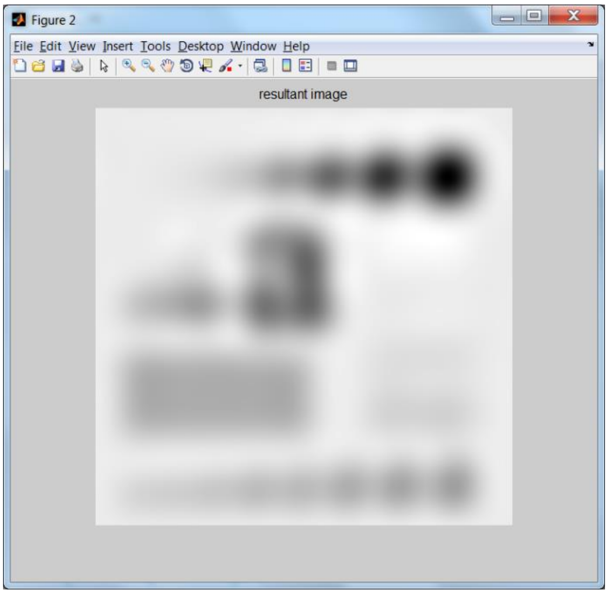


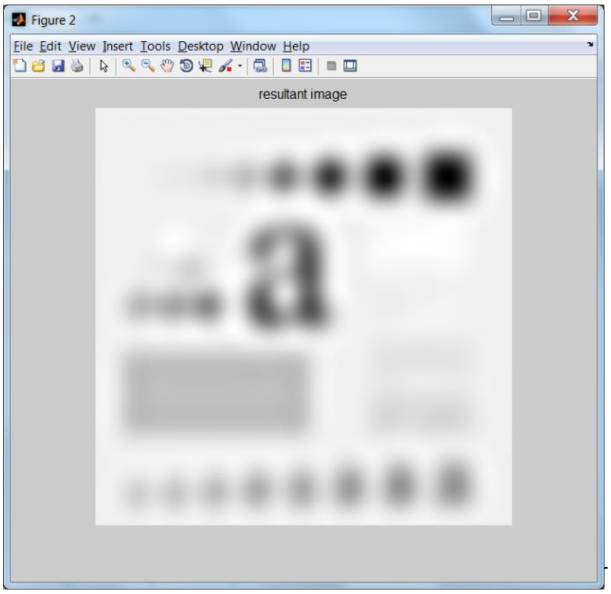


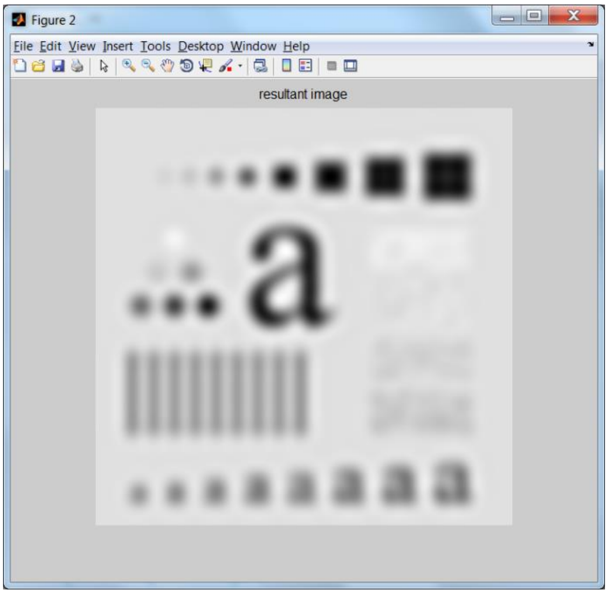
```
prac7_5('fig4.tif','lp','gaussian',3,3,5)
prac7_5('fig4.tif','lp','gaussian',3,3,20)
prac7_5('fig4.tif','lp','gaussian',3,3,35)
prac7_5('fig4.tif','lp','gaussian',3,3,75)
prac7_5('fig4.tif','lp','gaussian',3,3,225)
```



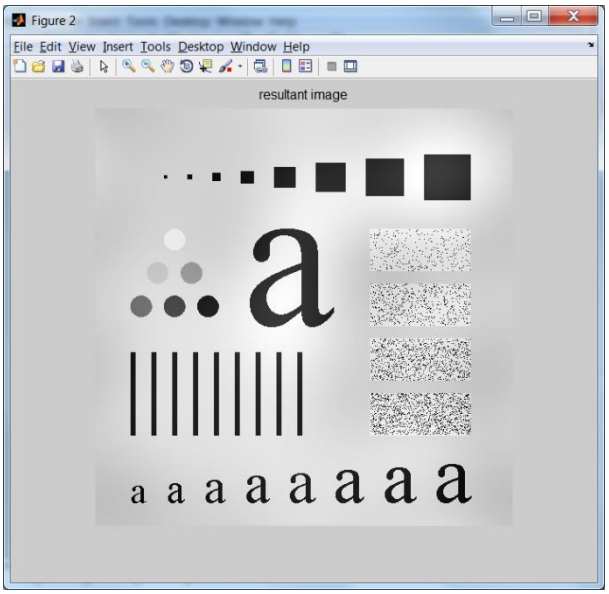


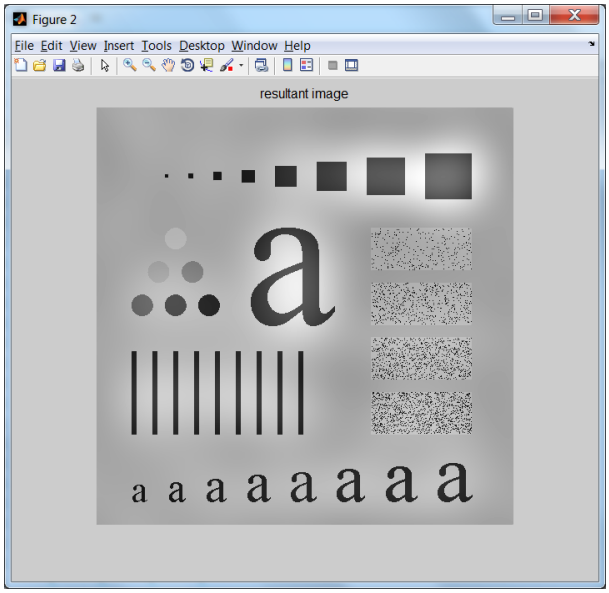


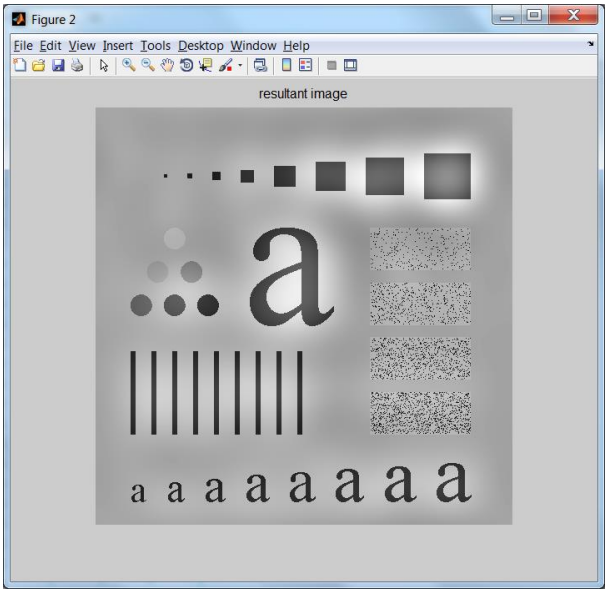


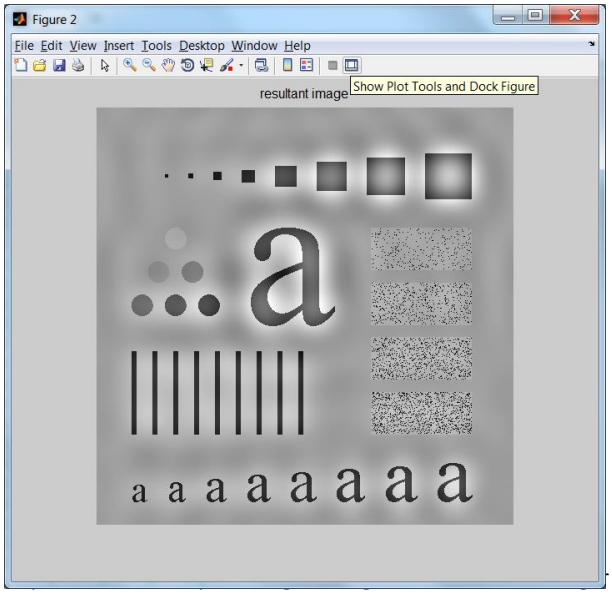


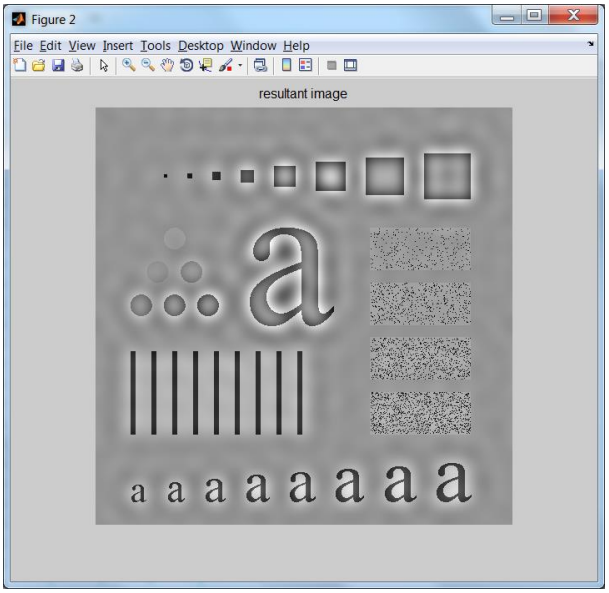
prac7_5('fig4.tif','hp','ideal',3,3,5,5)
prac7_5('fig4.tif','hp','ideal',3,3,20,5)
prac7_5('fig4.tif','hp','ideal',3,3,35,5)
prac7_5('fig4.tif','hp','ideal',3,3,75,5)
prac7_5('fig4.tif','hp','ideal',3,3,225,5)



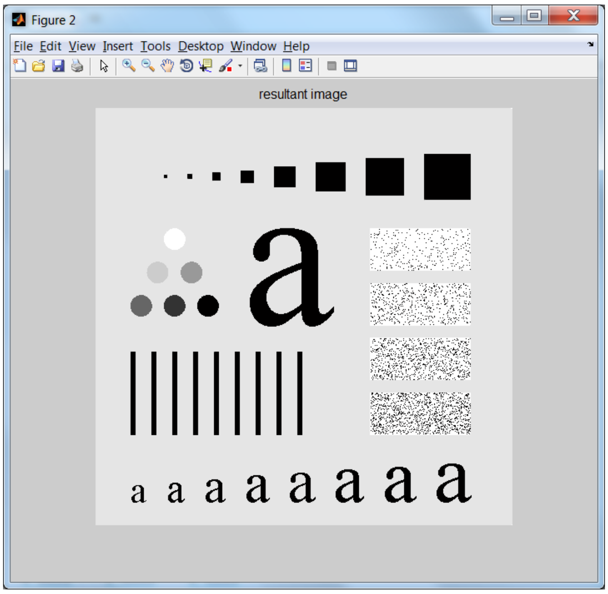


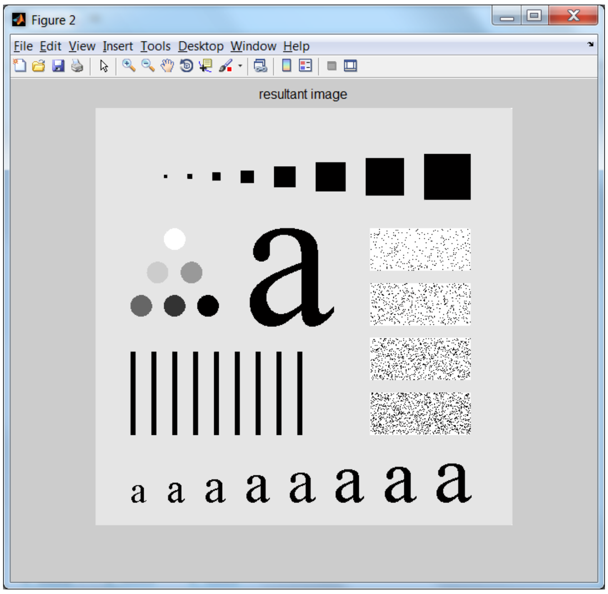


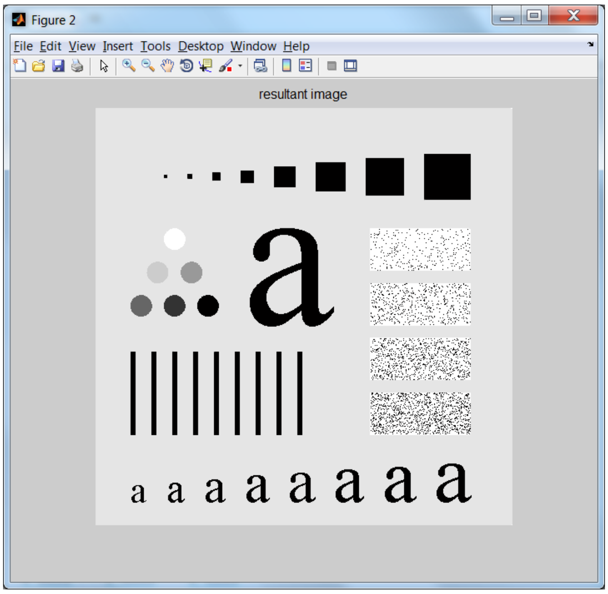


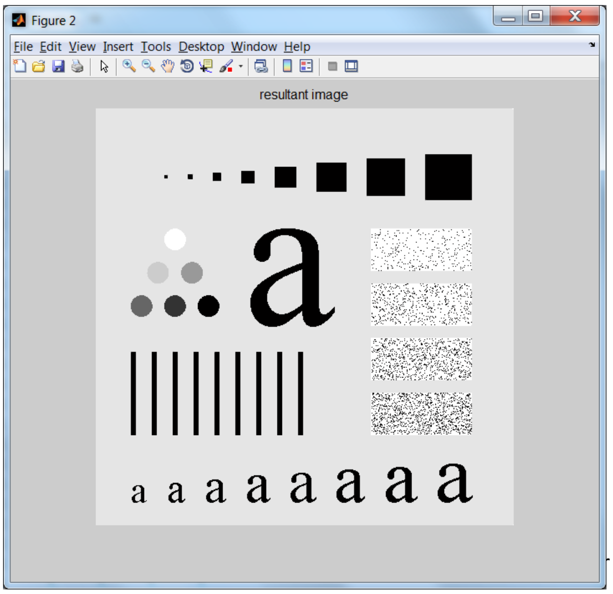


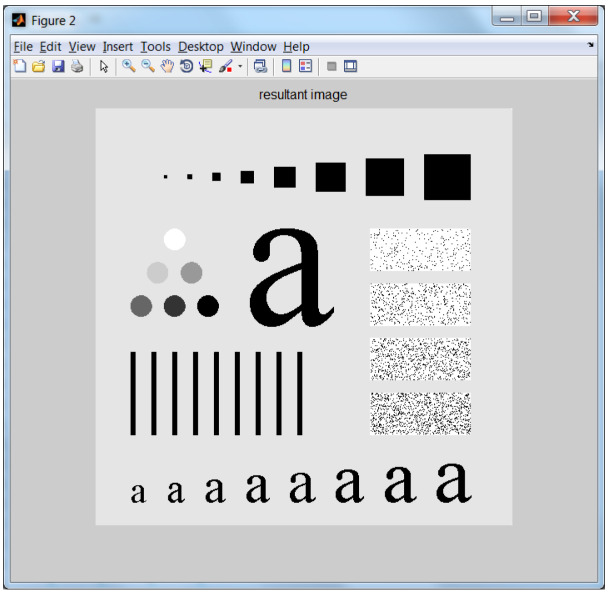
prac7_5('fig4.tif','hp','butterworth',3,3,5,5)
prac7_5('fig4.tif','hp','butterworth',3,3,20,5)
prac7_5('fig4.tif','hp','butterworth',3,3,35,5)
prac7_5('fig4.tif','hp','butterworth',3,3,75,5)
prac7_5('fig4.tif','hp','butterworth',3,3,225,5)



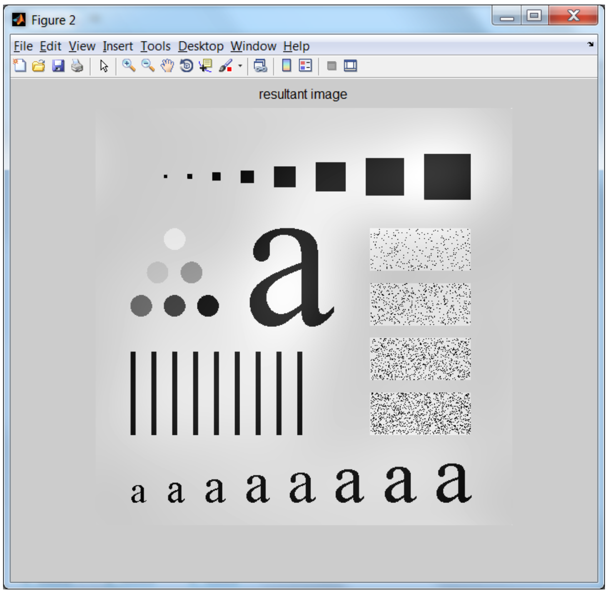


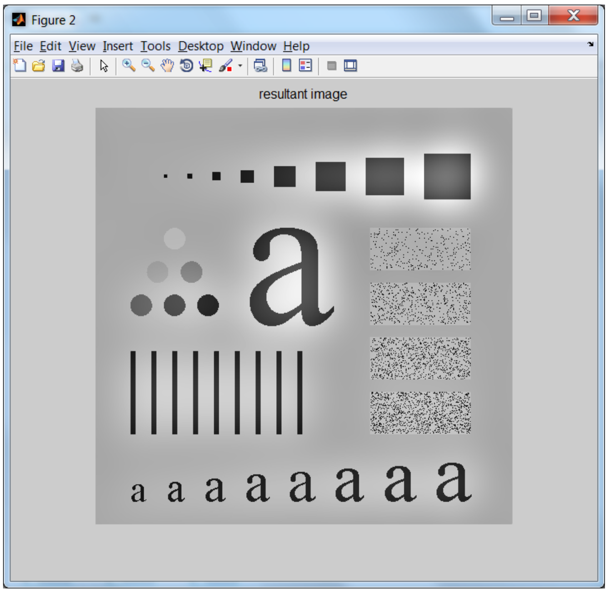


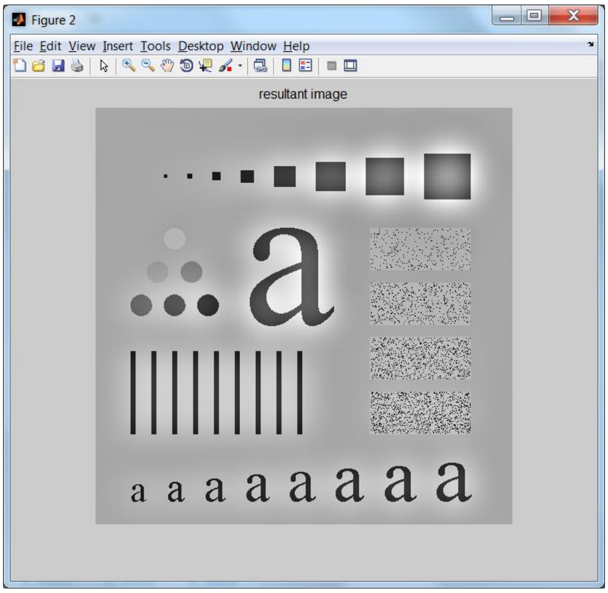


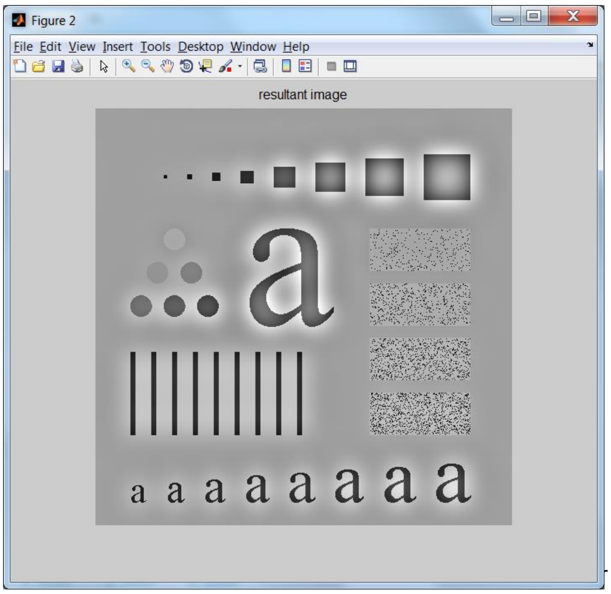


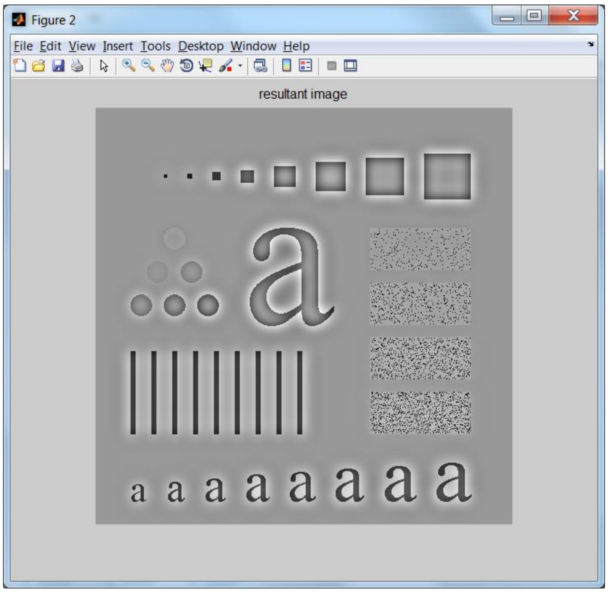
prac7_5('fig4.tif','hp','gaussian',3,3,5,5)
prac7_5('fig4.tif','hp','gaussian',3,3,20,5)
prac7_5('fig4.tif','hp','gaussian',3,3,35,5)
prac7_5('fig4.tif','hp','gaussian',3,3,75,5)
prac7_5('fig4.tif','hp','gaussian',3,3,225,5)











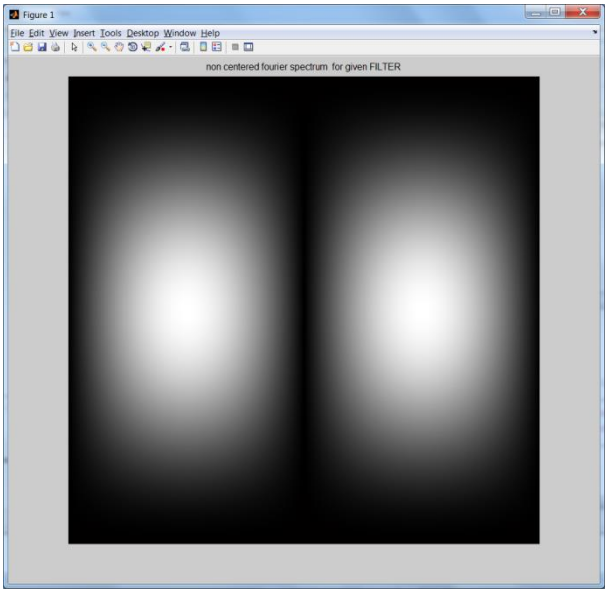
6. Generate sobel filter of size 1200 x 1200 in frequency domain corresponding to the vertical sobel filter of size 3 x 3 in the spatial domain. Display centered and non-centered fourier spectrum of the filter (In terms of wireframe plot and normal plot).

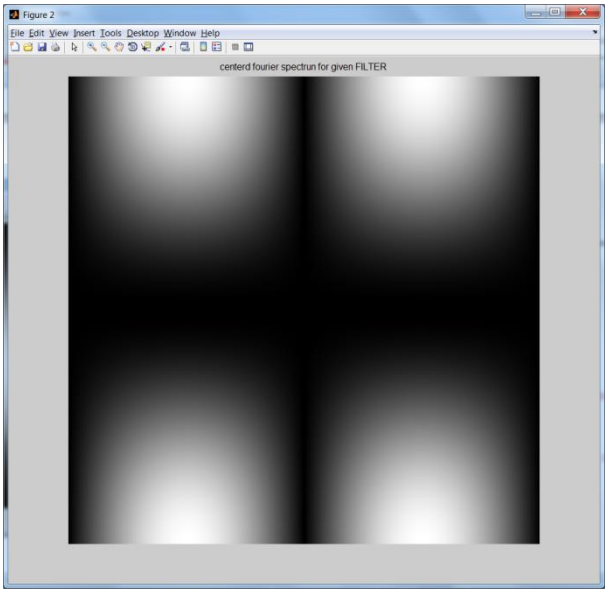
Program:

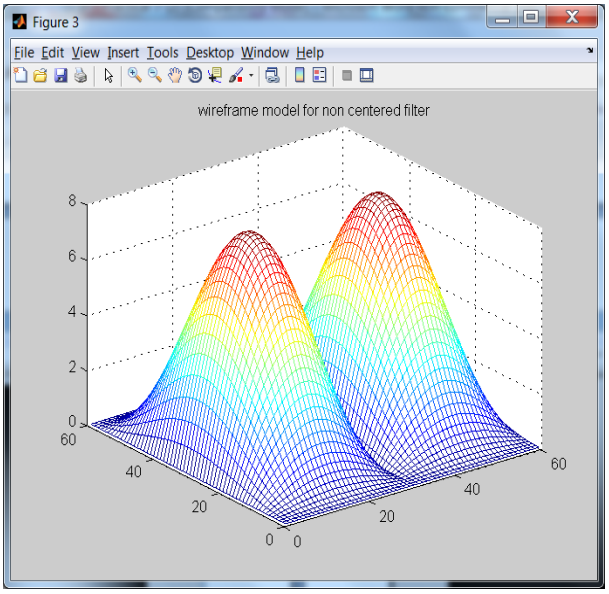
```
function prac7_6()
h=fspecial('sobel')';
h1=freqz2(h,[1200 1200]);
fs=abs(h1);%we are having absolute values in ft
figure();
imshow(fs,[]);
title('non centered fourier spectrum  for given FILTER');
%generate centered fourier spectrum
%fftshift will convert centered FILTER to non centered and vice versa
cfs=fftshift(fs);
figure();
imshow(cfs,[]);
title('centerd fourier spectrun for given FILTER');
%visually enhance fourier spectrum
[r c]=size(cfs);
figure();
mesh(fs(1:20:r,1:20:c));
title('wireframe model for non centered filter');
figure();
mesh(cfs(1:20:r,1:20:c));
title('wireframe model for centered filter');
end
```

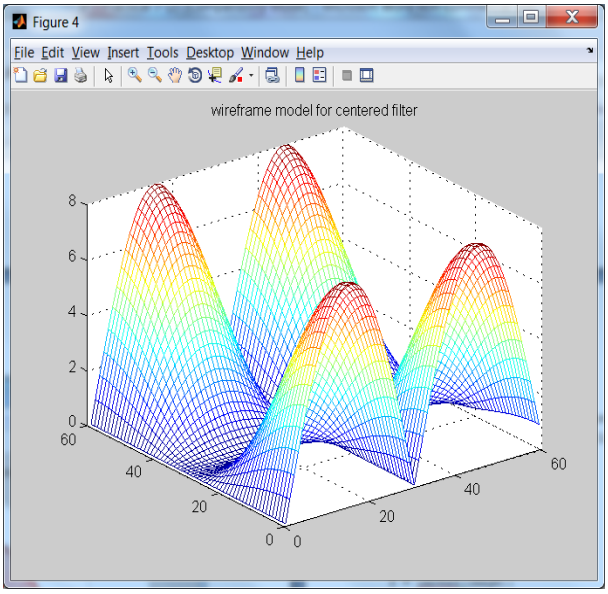
Output:

>> prac7_6









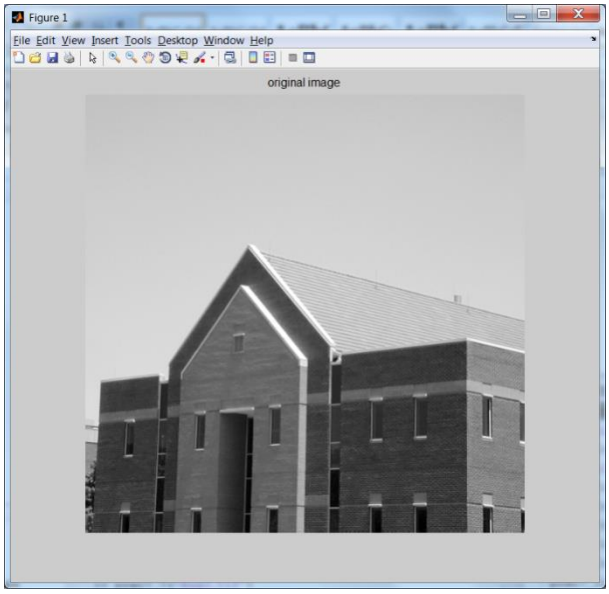
7. Apply vertical sobel filter (in spatial domain) of size 3 x 3 on the following image. Also apply corresponding frequency domain filter. Compare results.

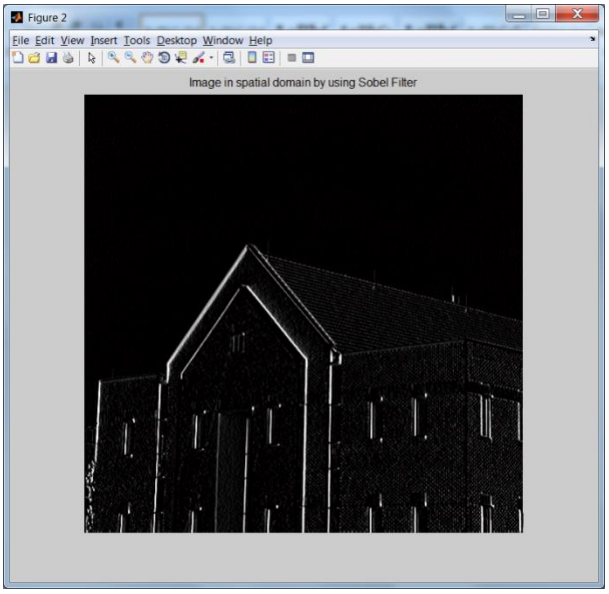
Program:

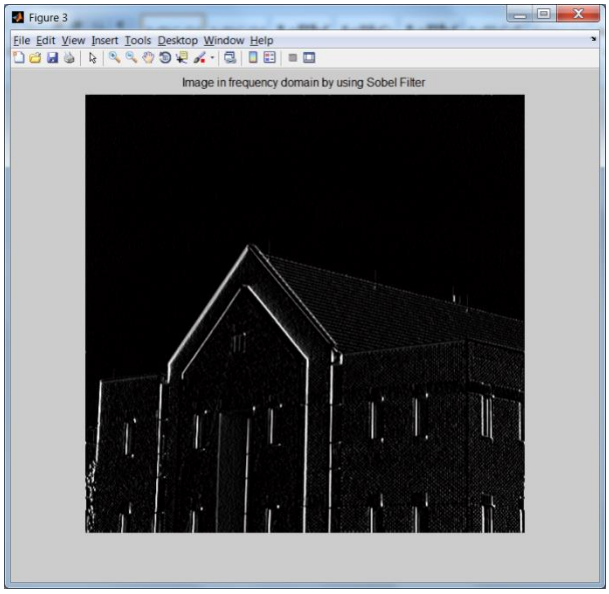
```
function prac7_7(image)
imshow(image);
title('original image');
f = imread(image);
[r c]=size(f);
h = fspecial('sobel')';
b = imfilter(f,h);
figure;
imshow(b);
title('Image in spatial domain by using Sobel Filter');
ft=fft2(f);
h1=freqz2(h,[r c]);
c=ft.*fftshift(h1);
c1=real(ifft2(c));
figure;
imshow(uint8(c1),[]);
title('Image in frequency domain by using Sobel Filter');
end
```

Output:

>>prac7_7('fig3.tif')





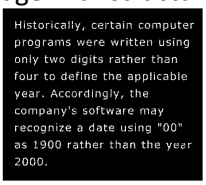


Practical Set-8

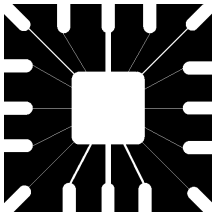
Tool: Matlab

- Consider following two images, A and B respectively. Display complement of A and B, A *union* B, A *intersection* B and A – B.

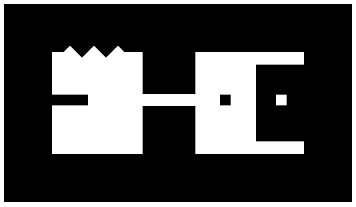
- Apply dilation on the following image with structuring element [0 1 0;1 1 1;0 1 0].



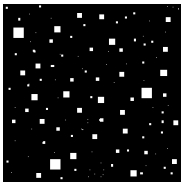
- Apply erosion on the following image. Use disk as a structuring element. Use radius of 5,10 and 20. Compare results.



- Apply opening and closing on the following image with structuring element of shape, square and width 20.



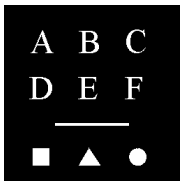
- Apply hit or miss transformation on the following image to accomplish the task of locating upper left corner pixel of each object in the following image.



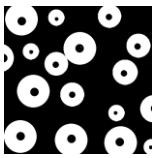
- Apply thinning on the following image. Also apply thinning on the resultant image. Compare results.

- Obtain the skeleton of the following image.

- Find out number of 4 and 8 connected components and number of pixels in each connected component in the following image.

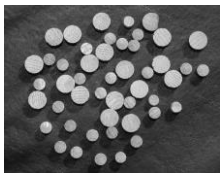


- Fill the black inner spots of all white circles in the following image.



- Apply dilation and erosion on the following gray scale image. Use square of width 3 as a structuring element.

- Apply opening and closing on the following grays scale image. Use disk of radius 5 as a structuring element.



Solution:

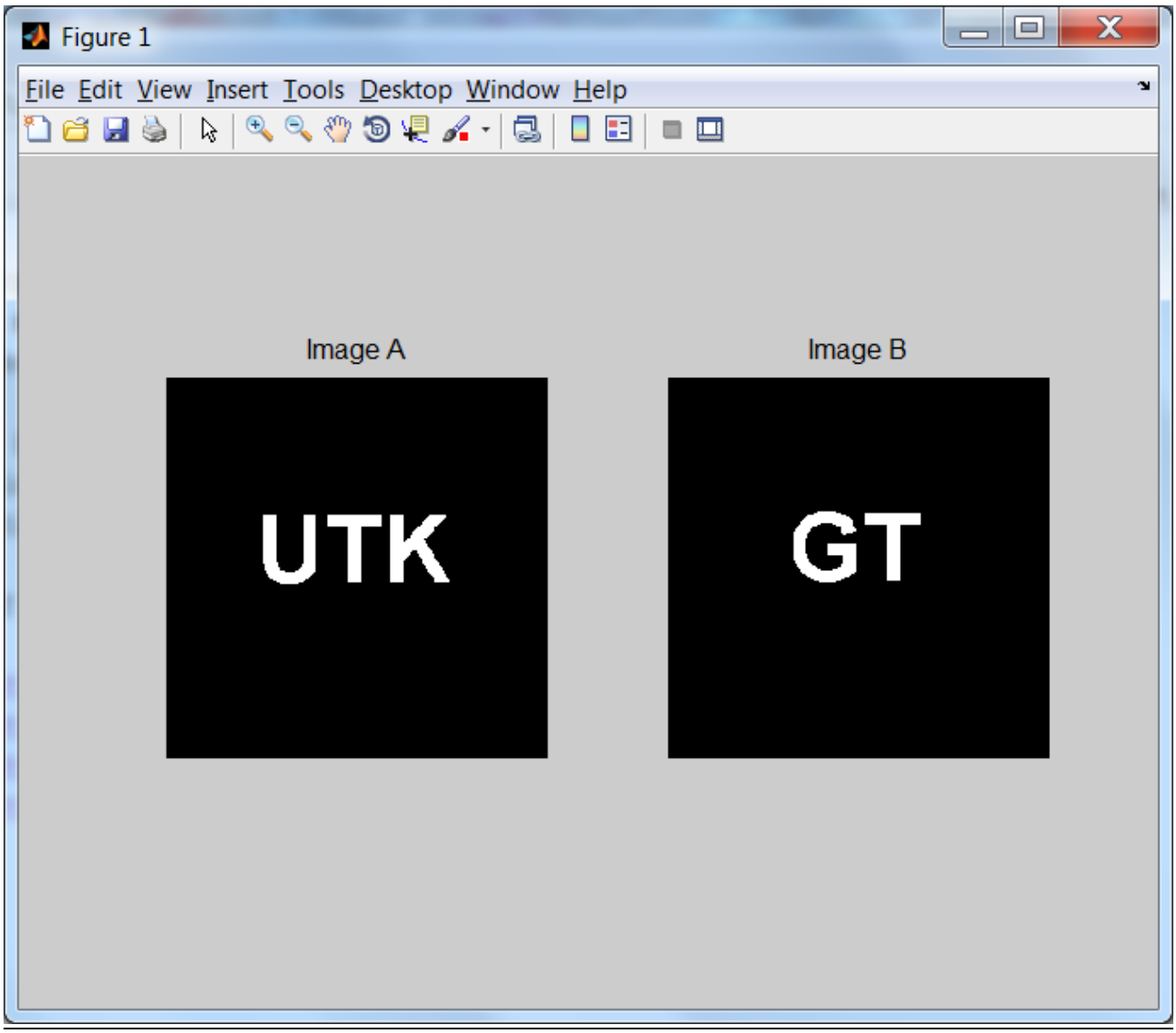
1)Consider following two images, A and B respectively. Display complement of A and B, A *union* B, A *intersection* B and A – B.

Program:

```
function parc8one(image1,image2)

% Function:complement of A and B, A union B, A intersection B and A – B.
% Description:This function display original images,complement of two
% images,complement of images,union of images,intersection of images,
% difference between A and B

subplot(1,2,1)
imshow(image1),title('Image A');
subplot(1,2,2)
imshow(image2),title('Image B');

a=imread(image1);
b=imread(image2);

coma=imcomplement(a);
comb=imcomplement(b);
figure;
subplot(1,2,1)
imshow(coma),title('Complement of Image A');
subplot(1,2,2)
imshow(comb),title('Complement of Image B');

uniab=a|b;
figure;
subplot(1,2,1)
imshow(uniab),title('Union of Image A & B');

interab=a&b;
subplot(1,2,2)
imshow(interab),title('Intersection of Image A & B');

diffab=a&comb;
figure;
imshow(diffab),title('difference of Image A & B');
end
```

output:

>> parc8one('fig1.tif','fig2.tif')



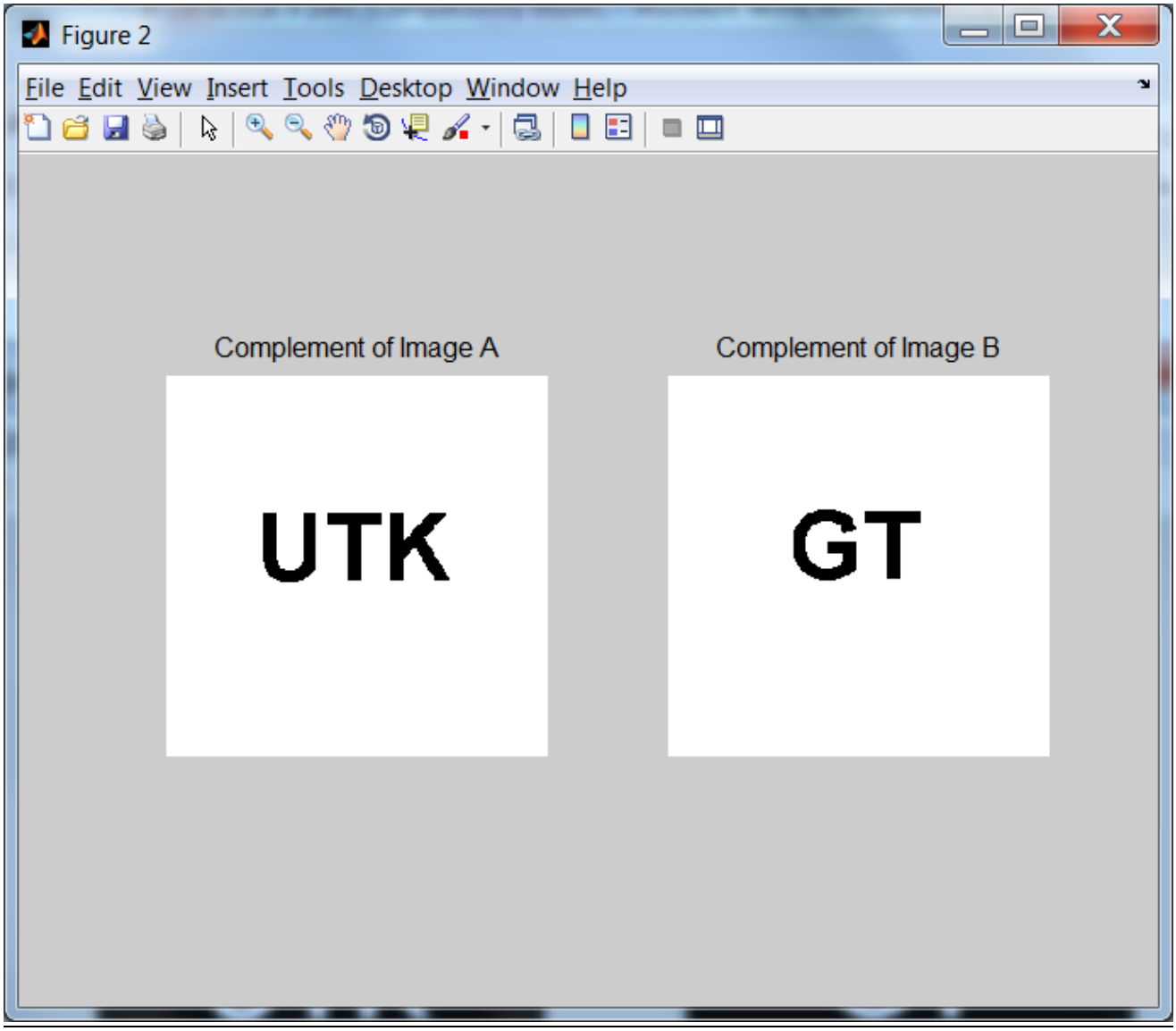


Figure 2
File Edit View Insert Tools Desktop Window Help
Complement of Image A
Complement of Image B
UTK
GT

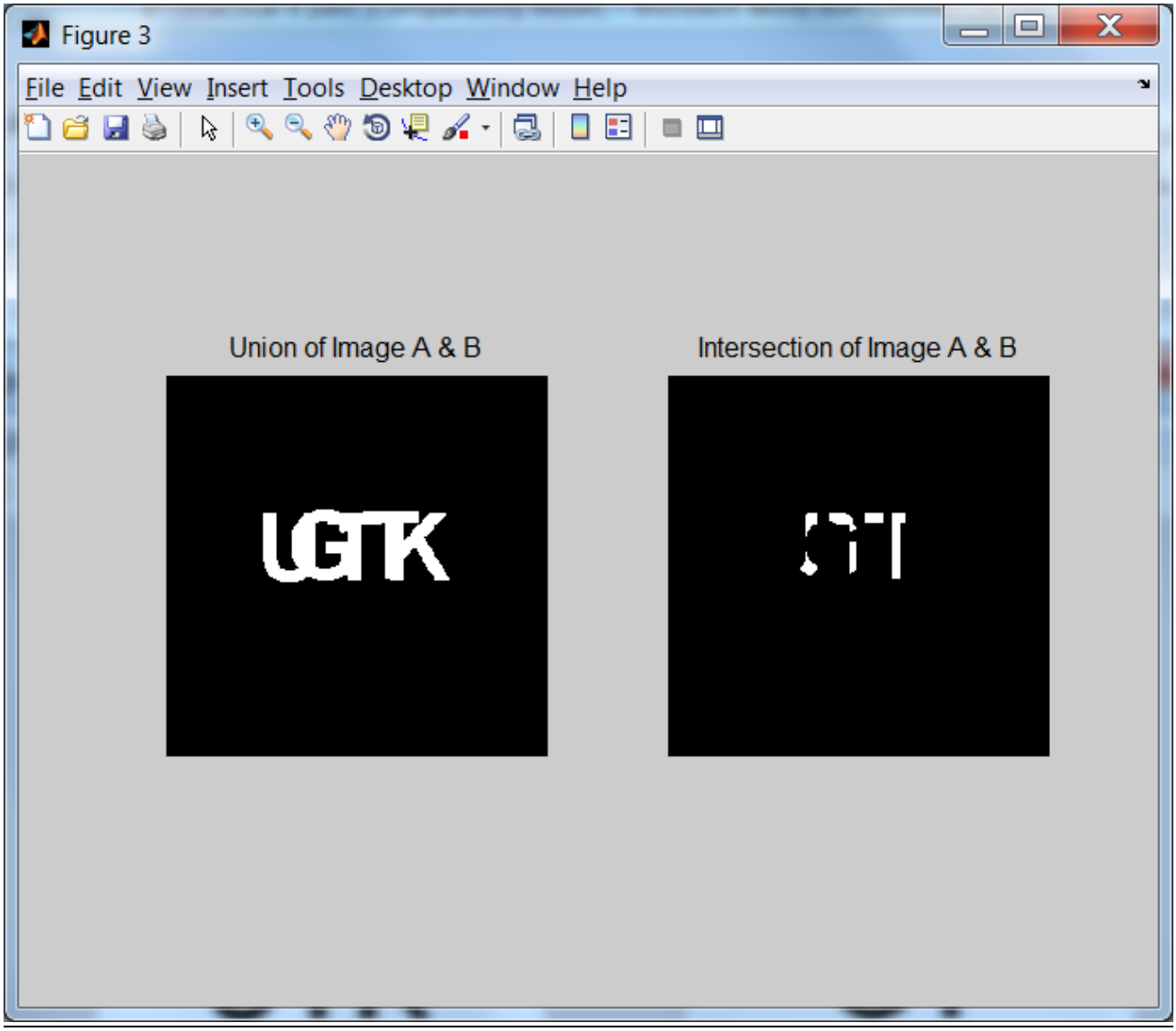
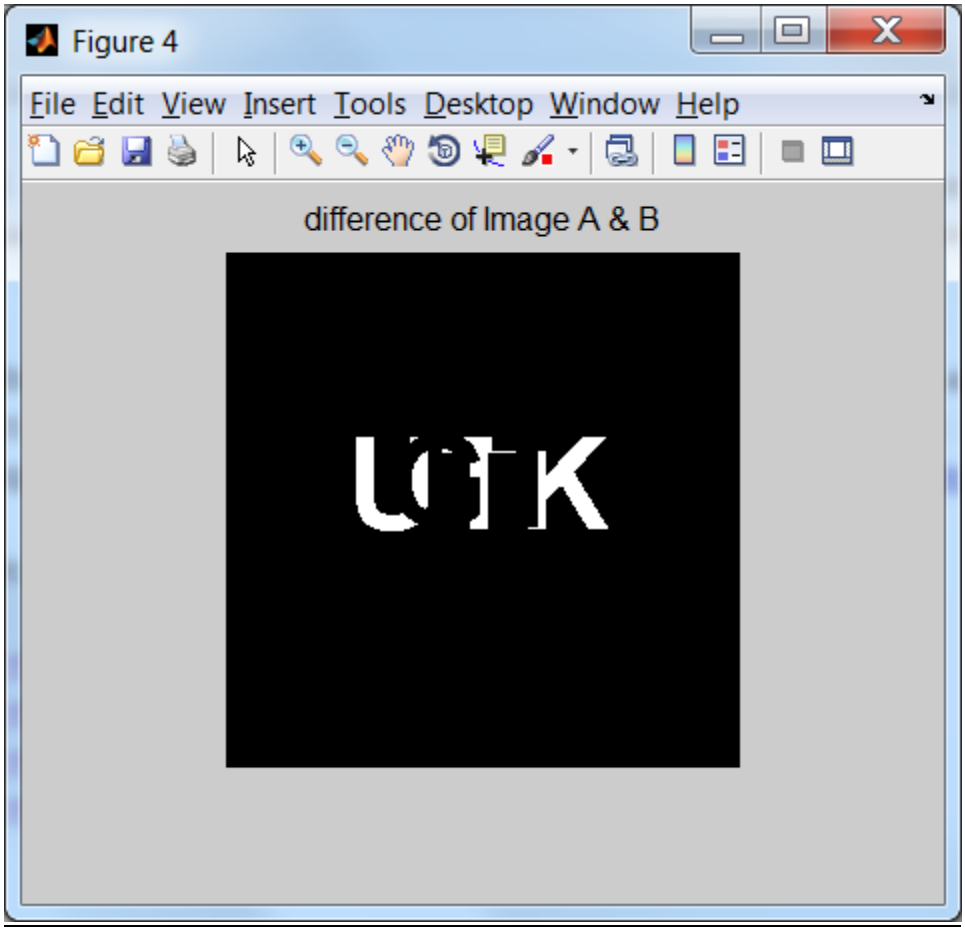
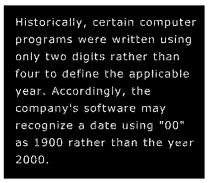


Figure 3
File Edit View Insert Tools Desktop Window Help
Union of Image A & B
Intersection of Image A & B



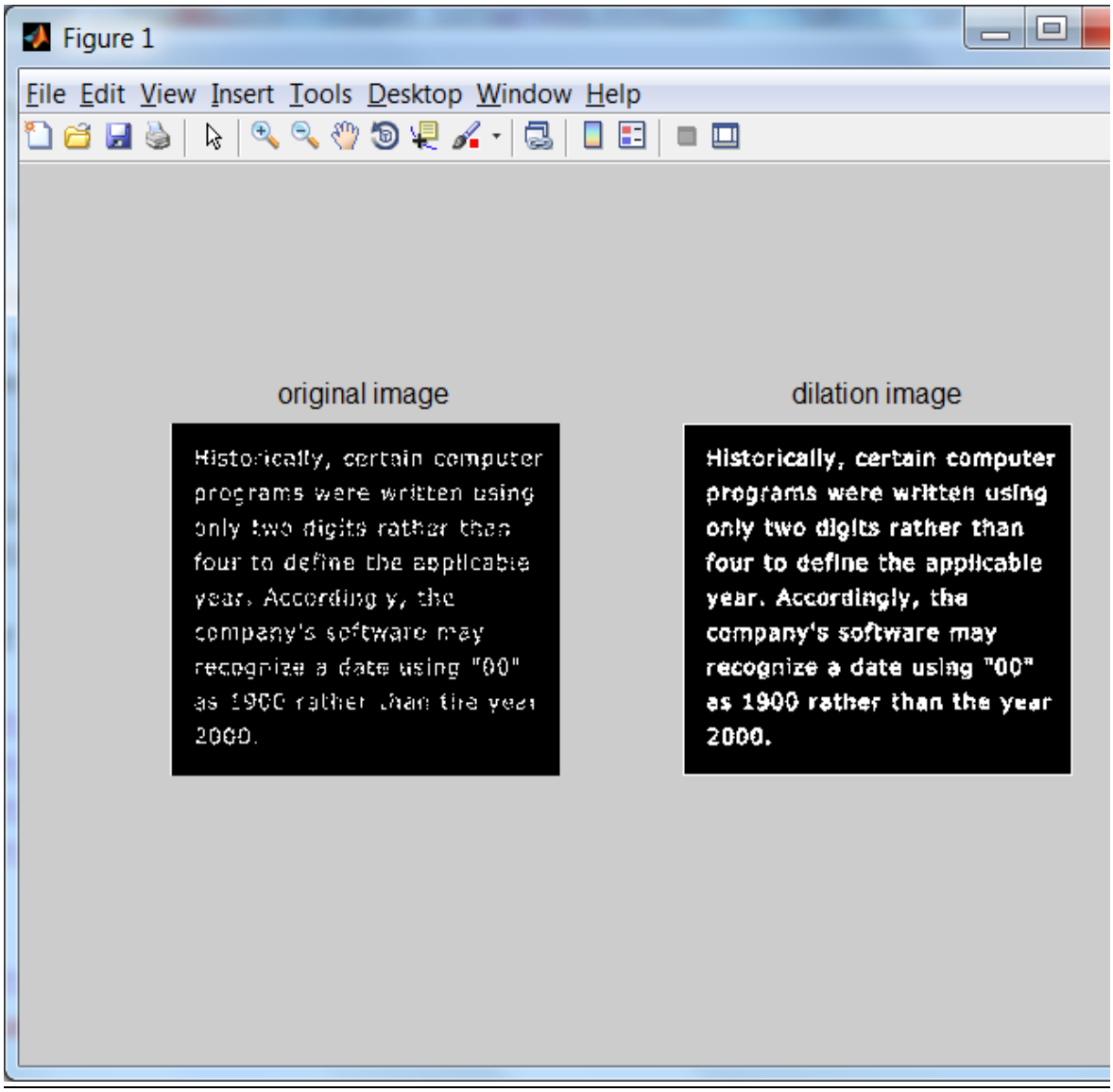
2)Apply dilation on the following image with structuring element [0 1 0;1 1 1;0 1 0].



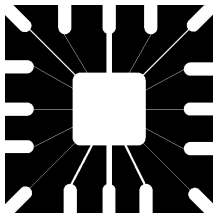

```
function parc8two(image)

% Function:Dilation
% Description:This function is use for apply dilation on the image
% with structuring element [0 1 0;1 1 1;0 1 0].

a=imread(image);
subplot(1,2,1)
imshow(a),title('original image');

s=[0 1 0;1 1 1;0 1 0];
d=imdilate(a,s);
subplot(1,2,2)
imshow(d),title('dilation image');
end
```

output:

>> parc8two('fig3.tif')



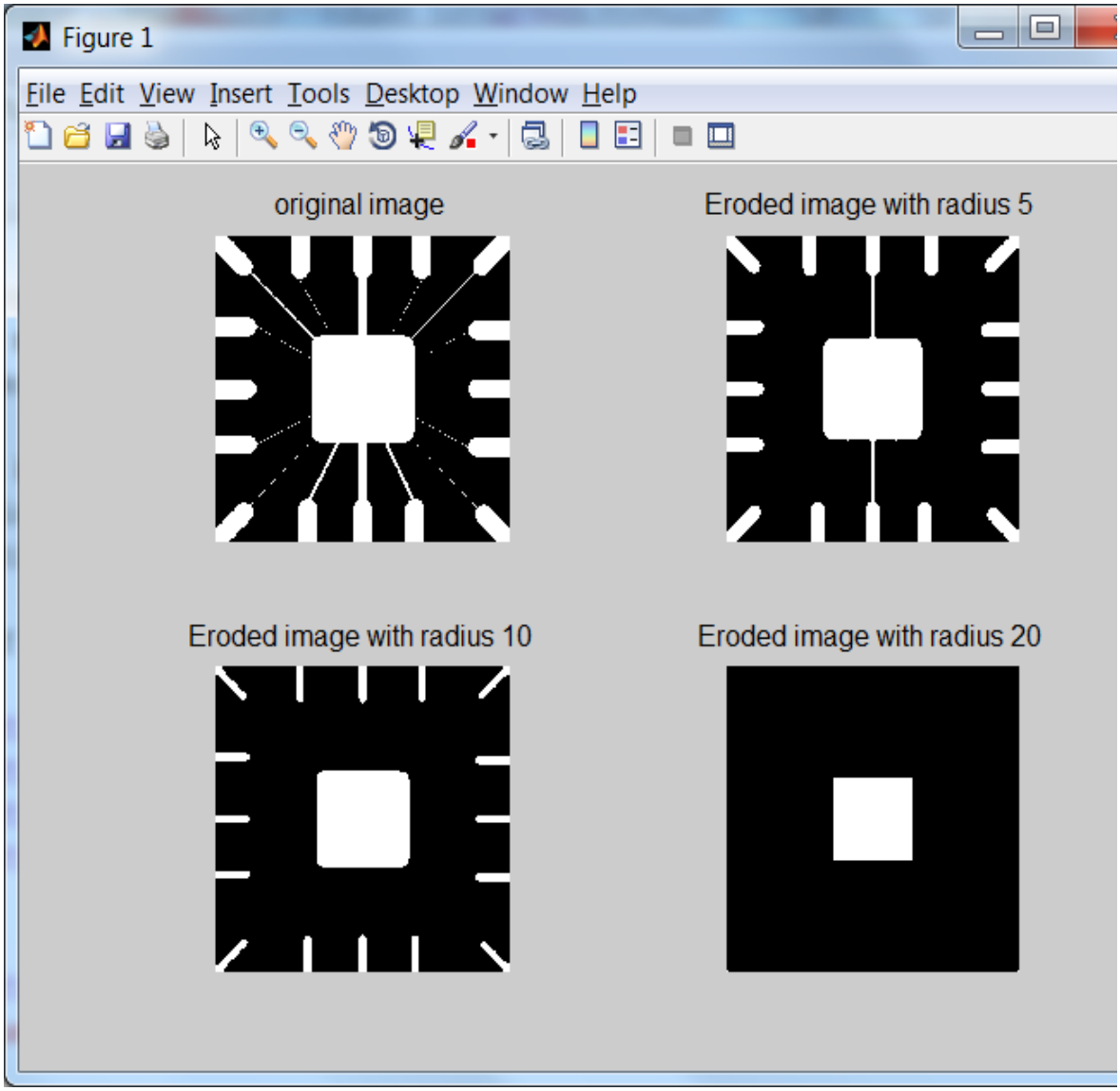
3)Apply erosion on the following image. Use disk as a structuring element. Use radius of 5,10 and 20. Compare results.



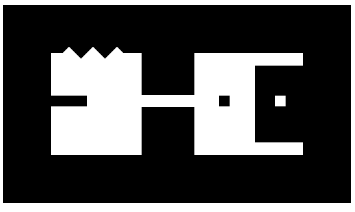
```
function parc8three(image)

% Function:Erosion
% Description:This function is use apply erosion on the image. Use disk
% as a structuring element. Use radius of 5,10 and 20. .

a=imread(image);
subplot(2,2,1)
imshow(a),title('original image');

s1=strel('disk',5);
e1=imerode(a,s1);
subplot(2,2,2)
imshow(e1),title('Eroded image with radius 5');

s2=strel('disk',10);
e2=imerode(a,s2);
subplot(2,2,3)
imshow(e2),title('Eroded image with radius 10');

s3=strel('disk',20);
e3=imerode(a,s3);
subplot(2,2,4)
imshow(e3),title('Eroded image with radius 20');
end
```

Output:

>> parc8three('fig4.tif')



4)Apply opening and closing on the following image with structuring element of shape, square and width 20.

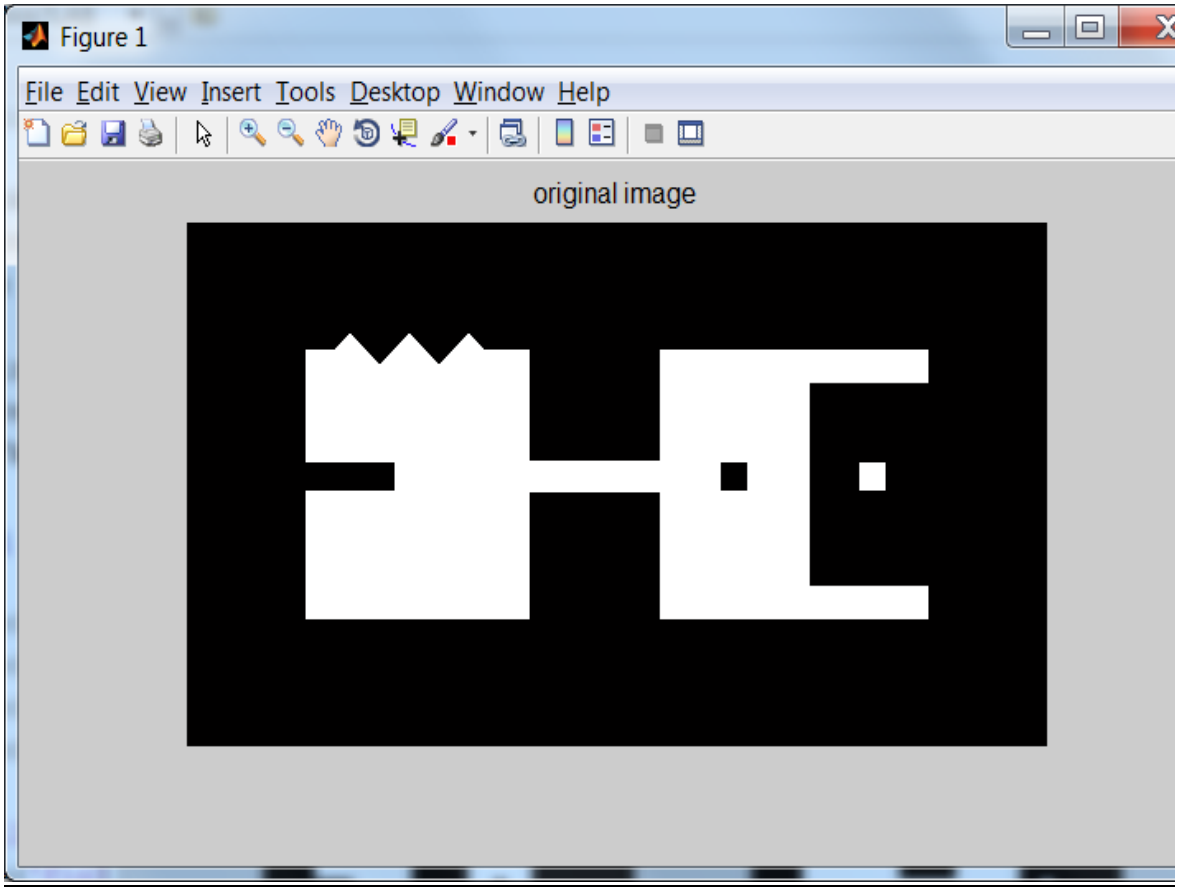


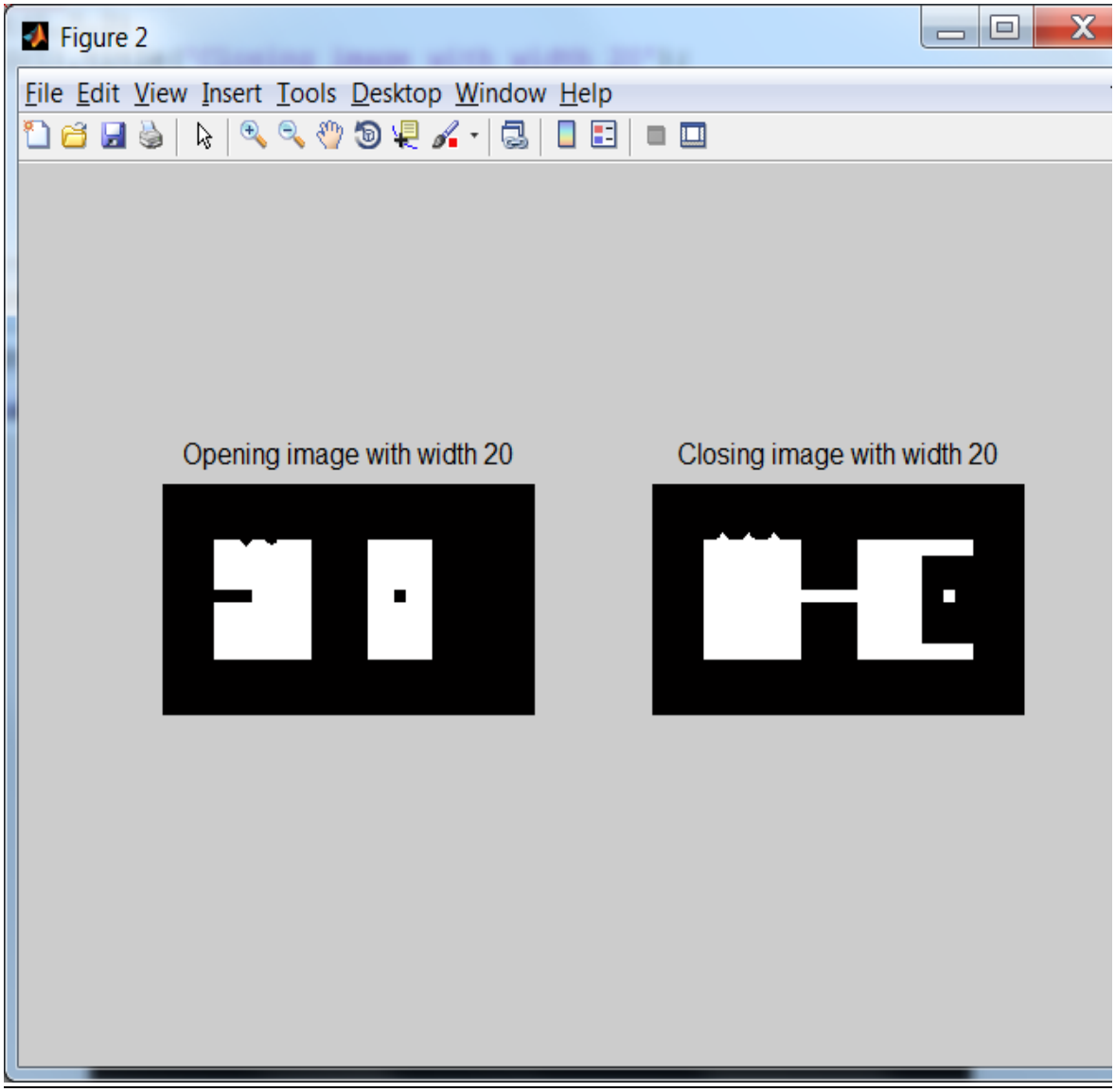
Program:

```
function parc8four(image)

% Function:Opening and closing
% Description:This function is use apply opening and closing on the
% image with structuring element of shape, square and width 20. .
```

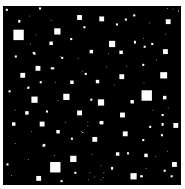
```
a=imread(image);
imshow(a),title('original image');
s1=strel('square',20);
o1=imopen(a,s1);

figure;
subplot(1,2,1)
imshow(o1),title('Opening image with width 20');

c1=imclose(a,s1);
subplot(1,2,2)
imshow(c1),title('Closing image with width 20');
end
```

output:





5)Apply hit or miss transformation on the following image to accomplish the task of locating upper left corner pixel of each object in the following image.



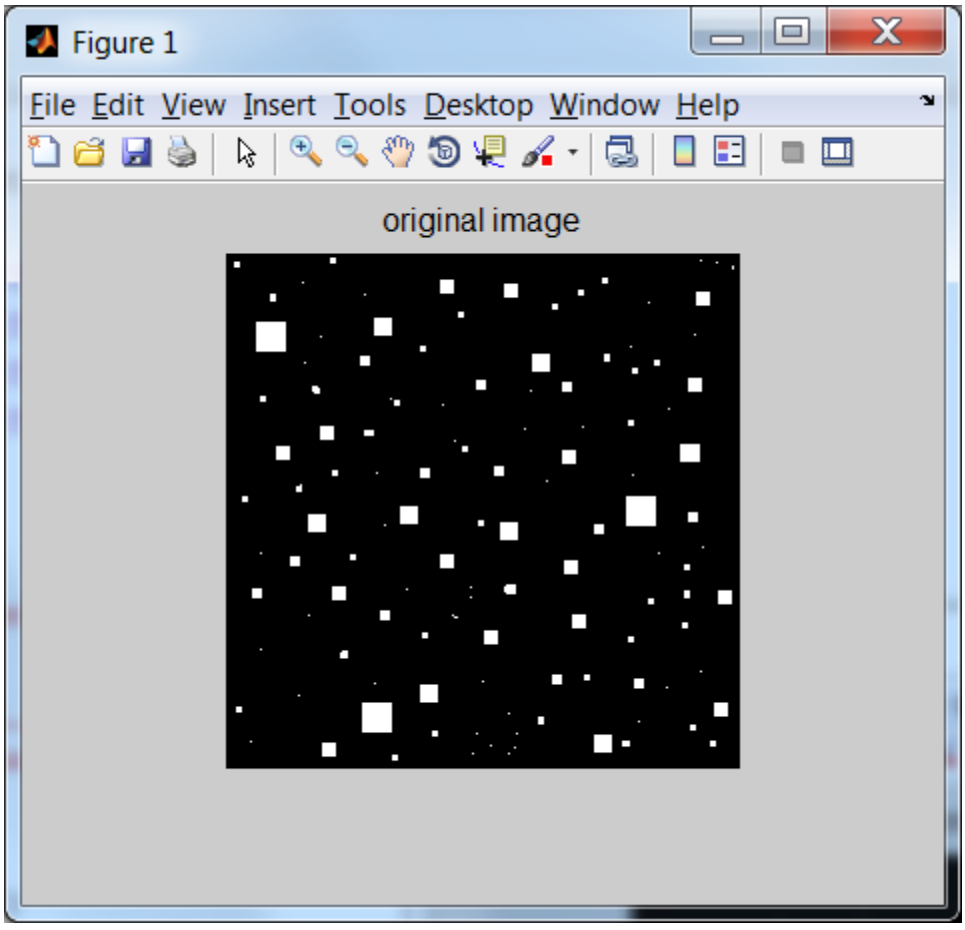
Program:

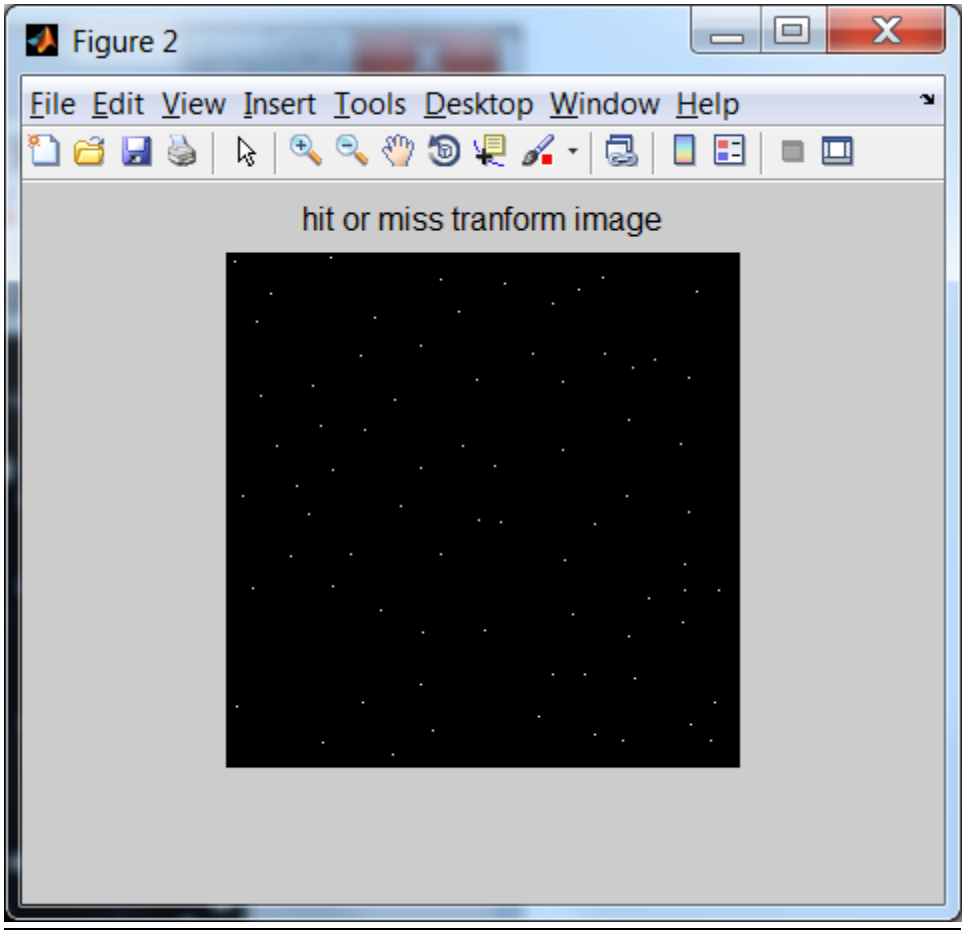
```
function parc8five(image)

% Function:Hit or Miss
% Description:This function is use Apply hit or miss transformation
on
% the following image to accomplish the task of locating upper left
% corner pixel of each object in the following image.
```

```
a=imread(image);
imshow(a),title('original image');

b1=[0 0 0;0 1 1;0 1 1];
b2=imcomplement(b1);
hm1=bwhitmiss(a,b1,b2);
figure;
imshow(hm1),title('hit or miss tranform image');
end
```

output:

```
>> parc8five('fig6.tif')
```





6)Apply thinning on the following image. Also apply thinning on the resultant image. Compare results.

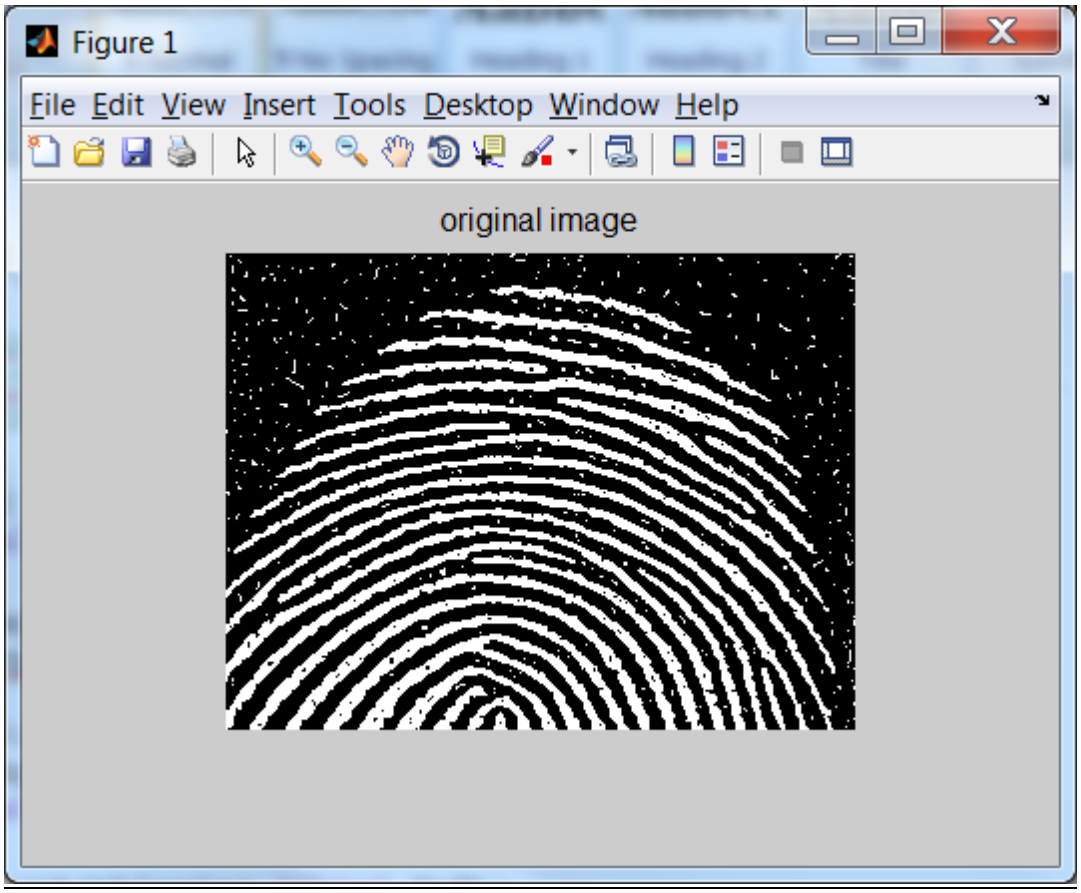
Program:

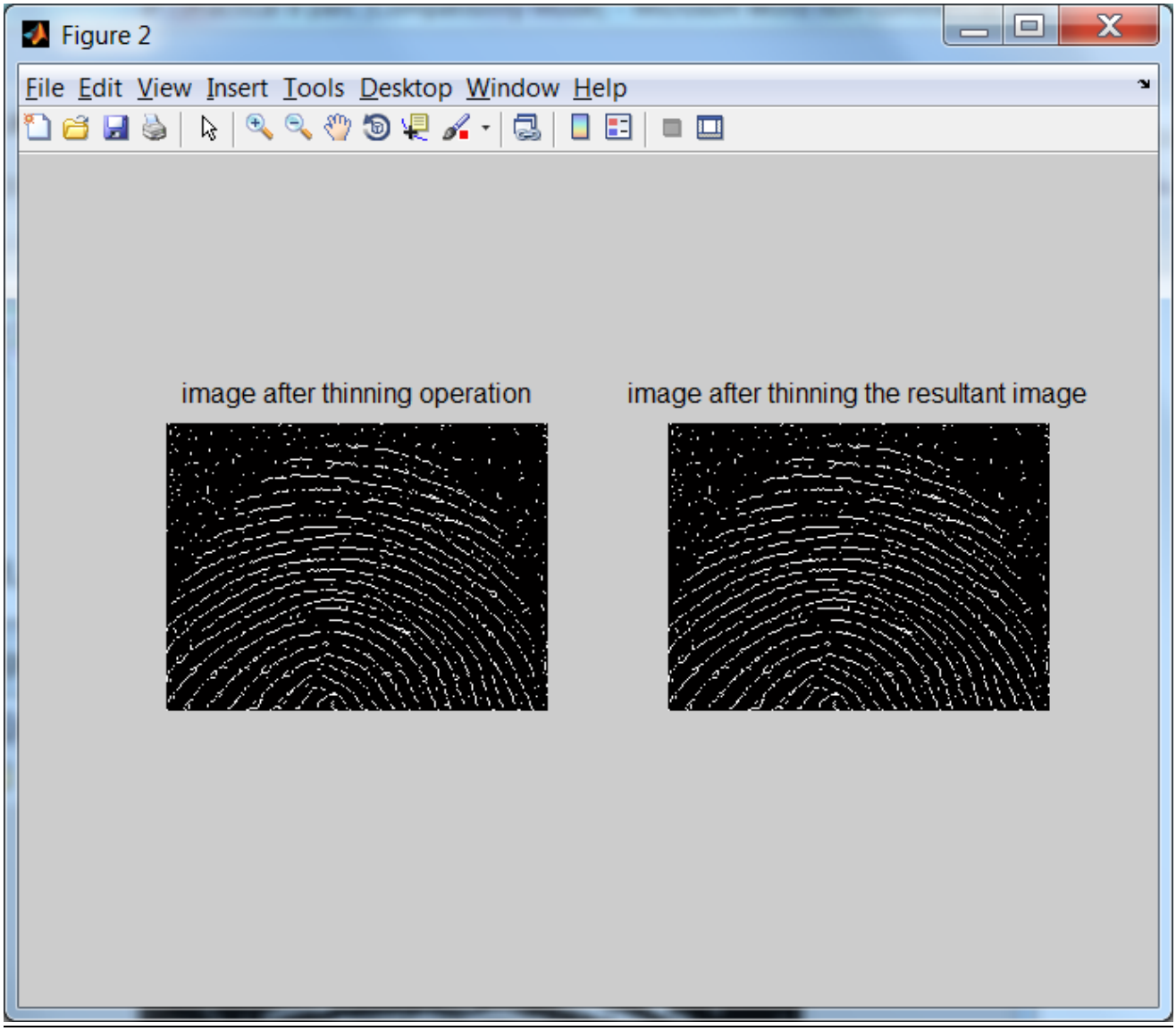
```
function prac8_6(image)

% Function:Thinning
% Description:This function is use for Apply thinning on the image.
% Also apply thinning on the resultant image.

imshow(image);
title('original image');

img=imread(image);
thinning=bwmorph(img,'thin',Inf);
```

```
figure();
subplot(1,2,1);
imshow(thinning);
title('image after thinning operation');

thinning1=bwmorph(thinning,'thin',Inf);
subplot(1,2,2);
imshow(thinning1);
title('image after thinning the resultant image');
end
```

output:

>> prac8_6('fig6.tif')





7)Obtain the skeleton of the following image.

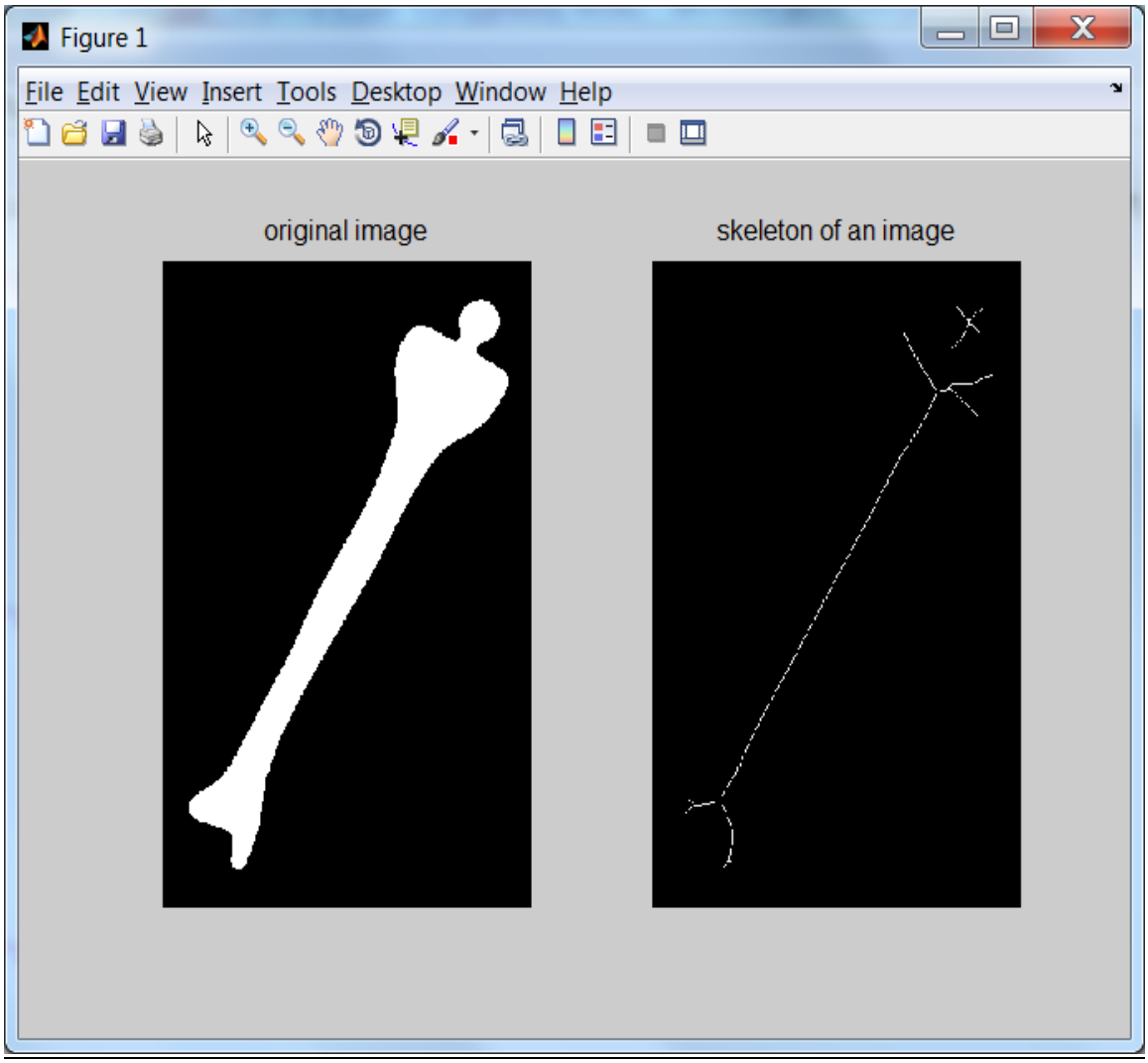
Program:

```
function prac8_7(image)

% Function:Skeleton
% Description:This function is use for  Obtain the skeleton of the
image

subplot(1,2,1);
imshow(image);
title('original image');
```

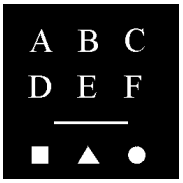
```
img=imread(image);
skeleton=bwmorph(img,'skel',Inf);
subplot(1,2,2);
imshow(skeleton);
title('skeleton of an image');
end
```

output:

>> prac8_7('fig8.tif')



8)Find out number of 4 and 8 connected components and number of pixels in each connected component in the following image.



Program:

```
function prac8_8(image)

% Function:Find Connected compoent
% Description:This function is use for  Find out number of 4 and 8
% connected components and number of pixels in each connected component
in
% the image.

a=imread(image);
[l,num]=bwlabel(a,8);
[l1,num1]=bwlabel(a,4);
sprintf('8 connected=%d 4-connected=%d',num,num1);
display('********8 connected pixel*********');
for i=1:num
    total=sum(sum(l==i));
    sprintf('Total number of pixels in connected component
%d=%d',i,total)
end
display('**********************************************************************
**');
display('********4 connected pixel*********');
for i=1:num1
    total=sum(sum(l==i));
    sprintf('Total number of pixels in connected component
%d=%d',i,total)

end
end
```

Output:

```
>> prac8_8('fig9.tif')

********8 connected pixel*********


ans =


Total number of pixels in connected component  1=396
```

ans =

Total number of pixels in connected component 2=260

ans =

Total number of pixels in connected component 3=600

ans =

Total number of pixels in connected component 4=424

ans =

Total number of pixels in connected component 5=301

ans =

Total number of pixels in connected component 6=400

ans =

Total number of pixels in connected component 7=408

ans =

Total number of pixels in connected component 8=260

ans =

Total number of pixels in connected component 9=251

ans =

Total number of pixels in connected component 10=490

****

********4 connected pixel*********

ans =

Total number of pixels in connected component 1=396

ans =

Total number of pixels in connected component 2=260

ans =

Total number of pixels in connected component 3=600

ans =

Total number of pixels in connected component 4=424

ans =

Total number of pixels in connected component 5=301

ans =

Total number of pixels in connected component 6=400

ans =

Total number of pixels in connected component 7=408

ans =

Total number of pixels in connected component 8=260

ans =

Total number of pixels in connected component 9=251

ans =

Total number of pixels in connected component 10=490

ans =

Total number of pixels in connected component 11=0

ans =

Total number of pixels in connected component 12=0

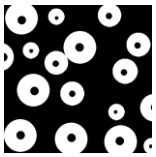
ans =

Total number of pixels in connected component 13=0

ans =

Total number of pixels in connected component 14=0

9)Fill the black inner spots of all white circles in the following image.

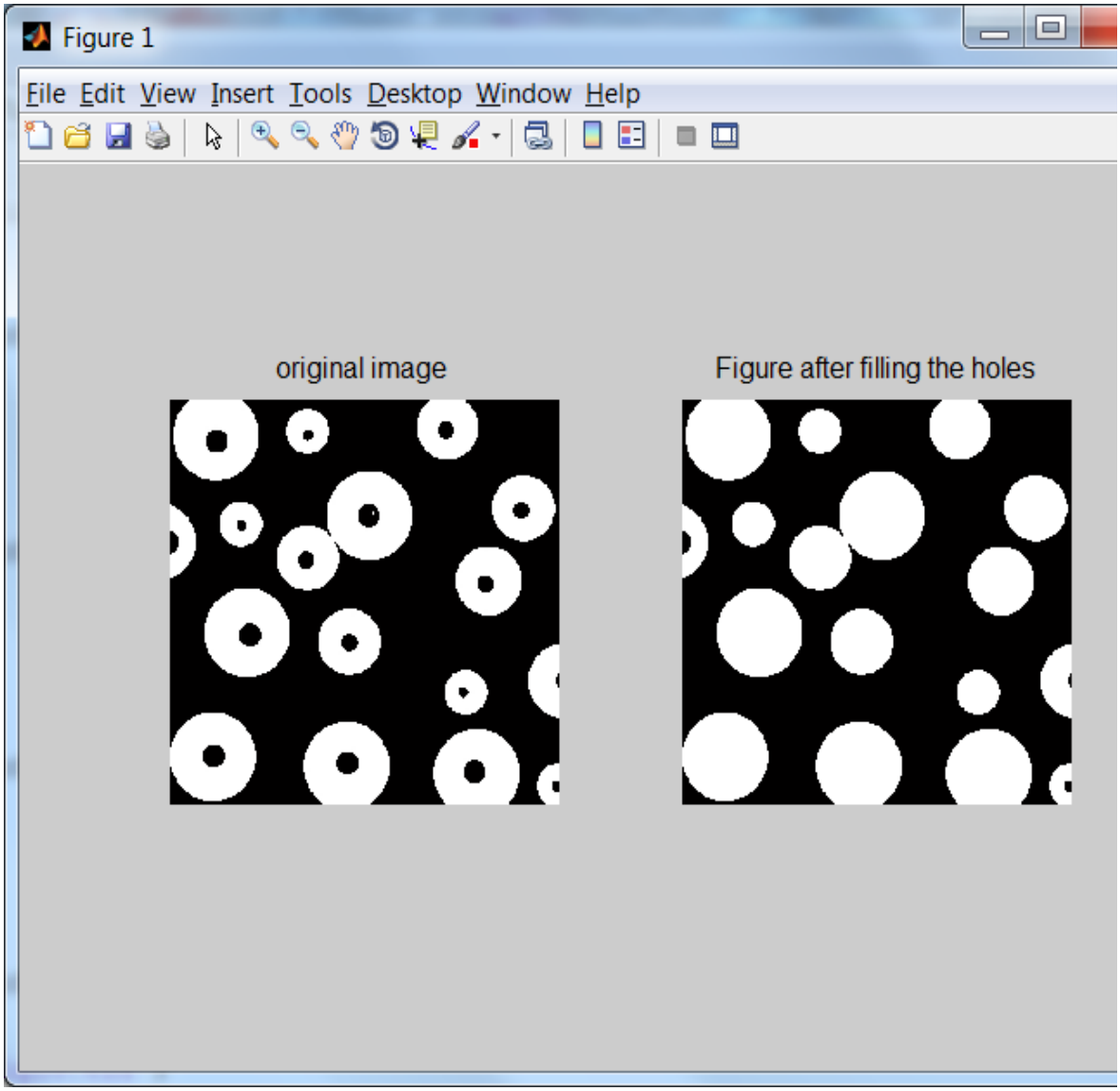


Program:

```
function prac8_9(image)

% Function:Fill Black inner spot
% Description:This function is use for  Fill the black inner spots of
all
% white circles in the following image.

subplot(1,2,1);
imshow(image);
title('original image');

a=imread(image);
b=imfill(a,'holes');
subplot(1,2,2);
imshow(b);
title('Figure after filling the holes');
end
```

Output:

>> prac8_9('fig10.jpg')



10)Apply dilation and erosion on the following gray scale image. Use square of width 3 as a structuring element.

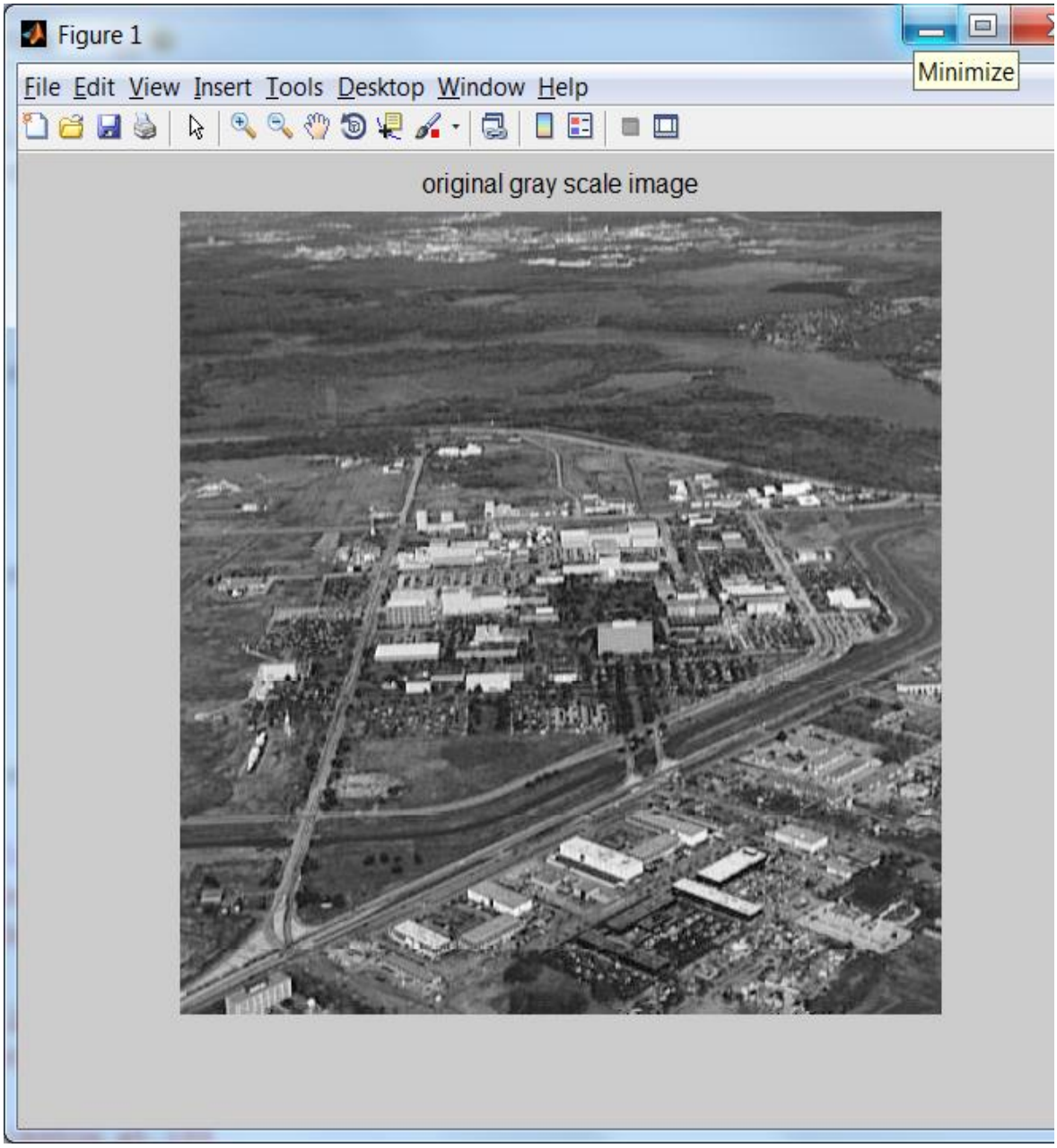
Program:

```
function prac8_10(image)

% Function:Erosion and Dilation on Gray scale image
% Description:This function is use for apply dilation and erosion on the
% gray scale image. Use square of width 3 as a structuring element.

imshow(image);
title('original gray scale image');
a=imread(image);
```

```
se=strel('square',3);
erosion=imerode(a,se);
figure();
imshow(erosion);
title('Gray scale image after erosion');
dilation=imdilate(a,se);
figure();
imshow(dilation);
title('Gray scale image after dilation');
end
```

output:

>> prac8_10('fig11.tif')



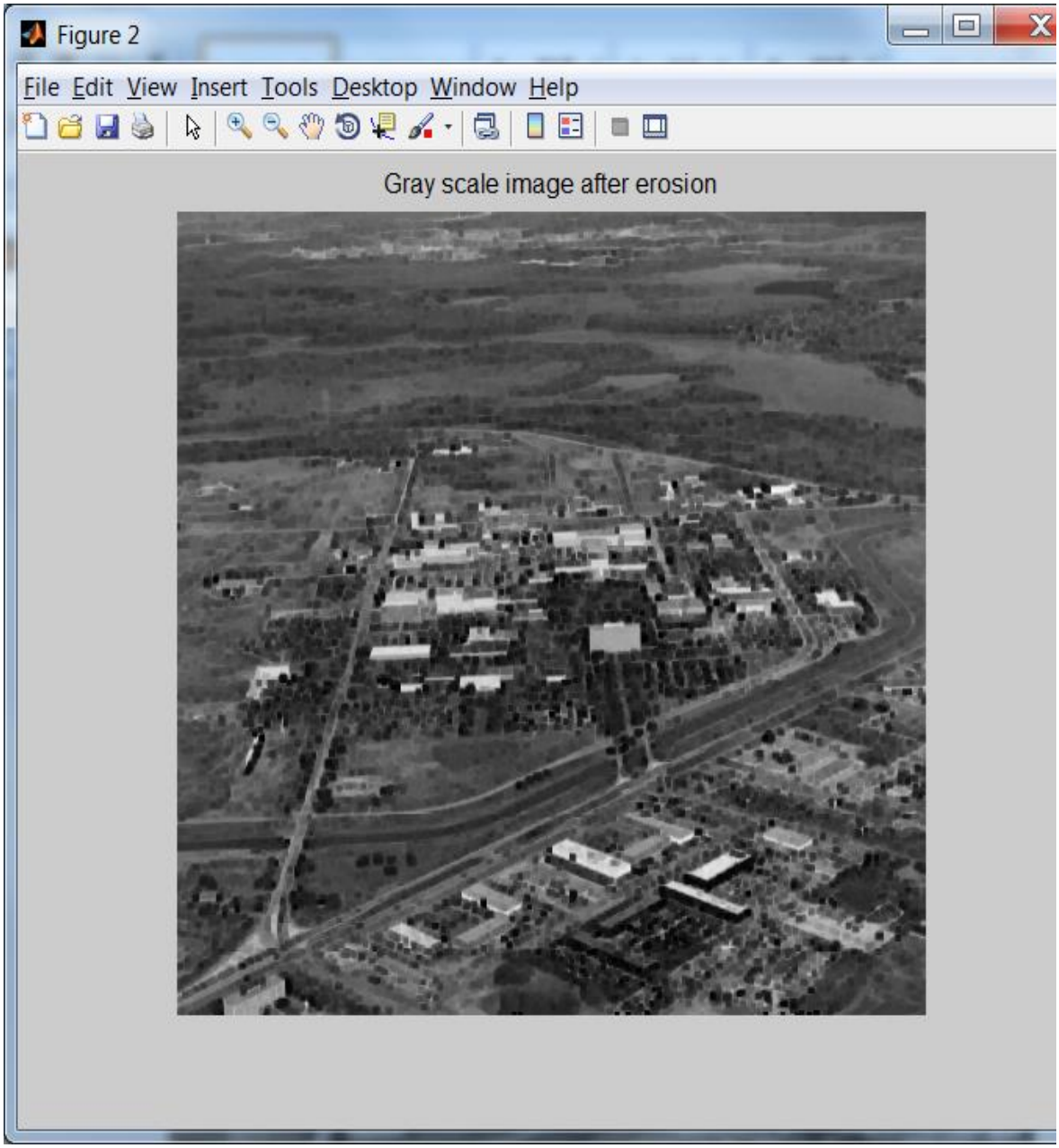
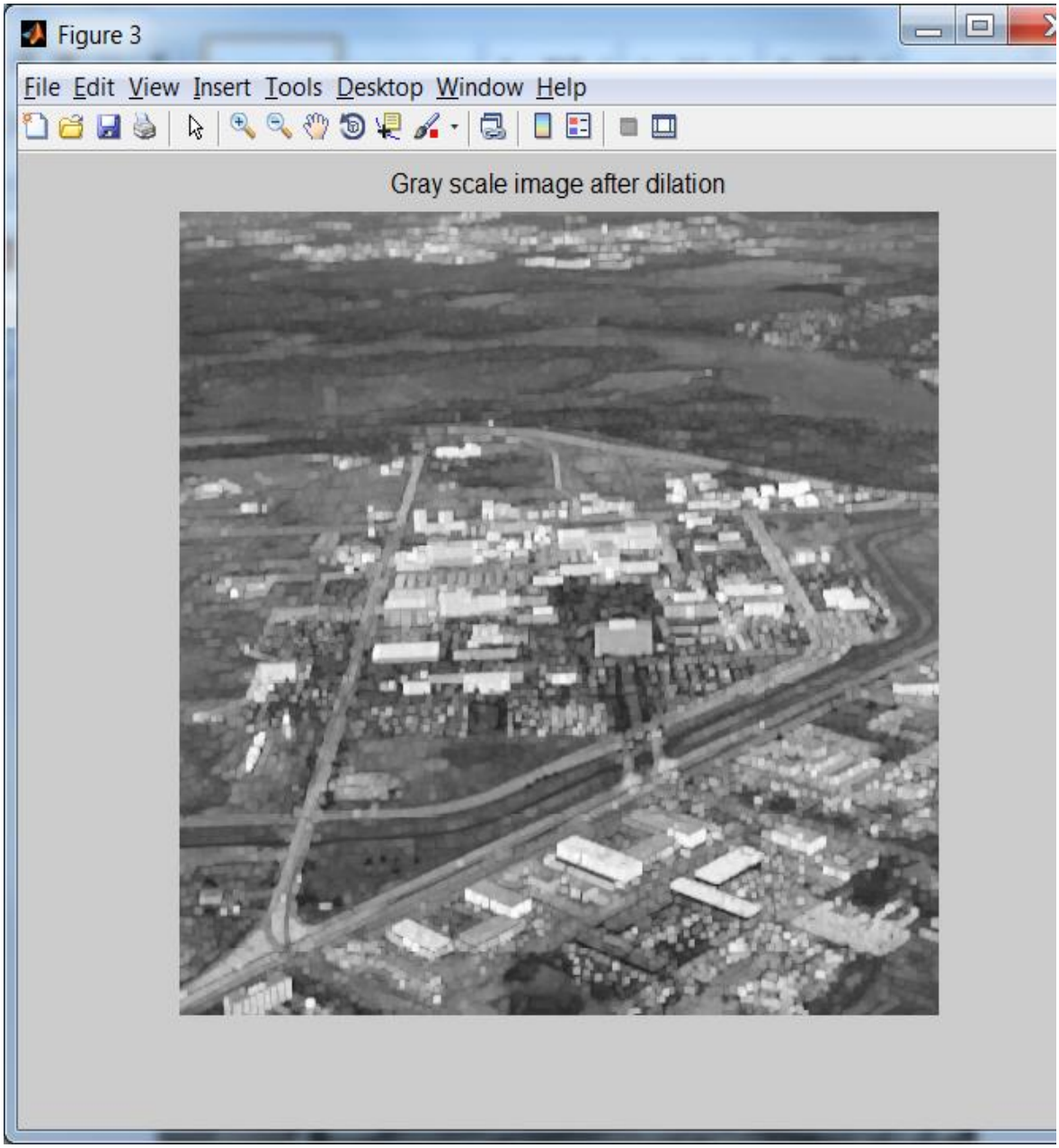


Figure 2
File Edit View Insert Tools Desktop Window Help
Gray scale image after erosion



11)Apply opening and closing on the following grays scale image. Use disk of radius 5 as a structuring element.

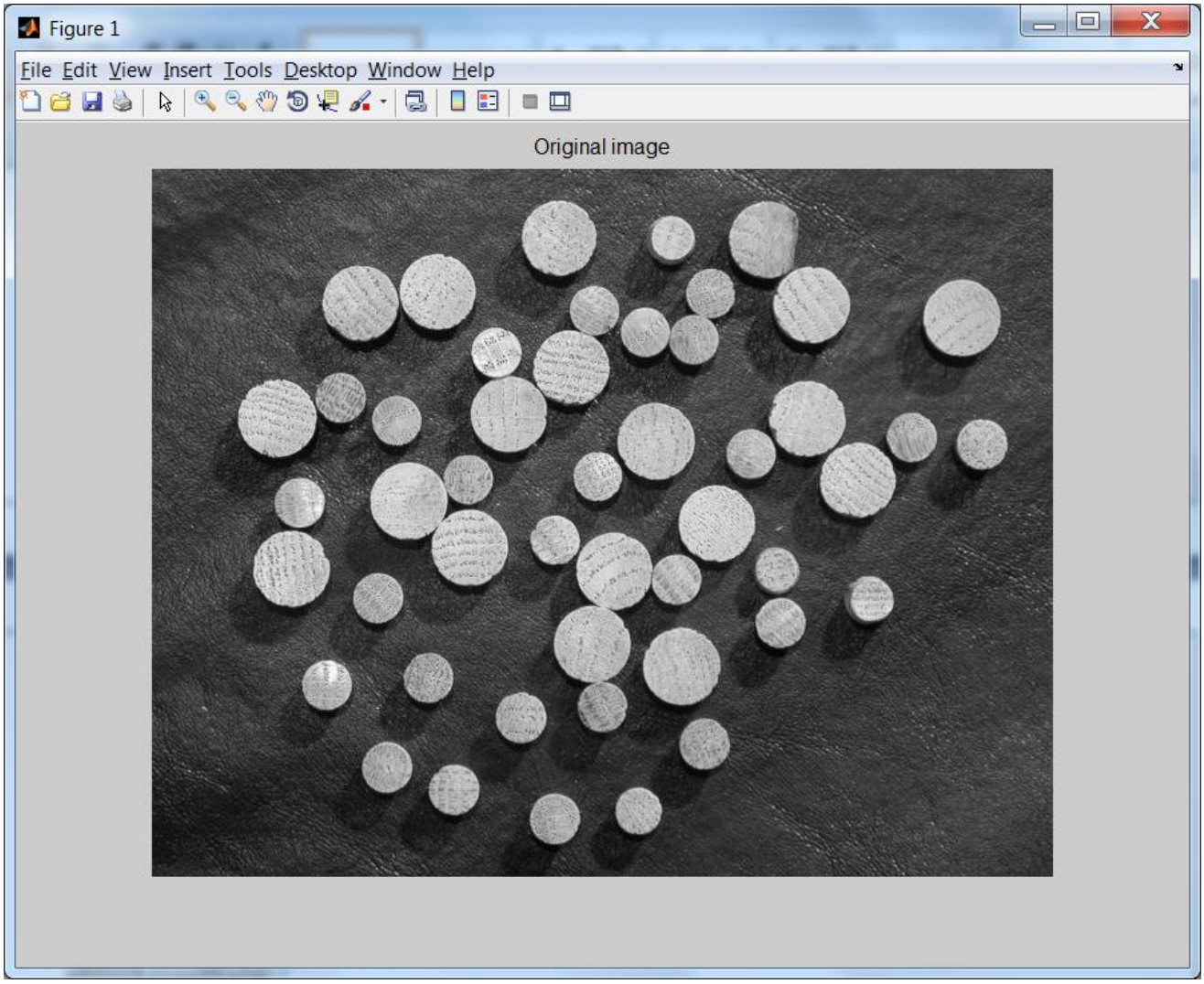
Program:

```
function prac8_11(image)

% Function:opening and closing on Gray scale image
% Description:This function is use for apply opening and closing on the
```

```
% grays scale image. Use disk of radius 5 as a structuring element.

imshow(image);
title('Original image');
a=imread(image);
se=strel('disk',5);

opening=imopen(a,se);
figure();
imshow(opening);
title('Gray scale image after opening');

closeing=imclose(a,se);
figure();
imshow(closeing);
title('Gray scale image after closing');
end
```

output:

>> prac8_11('fig12.tif')



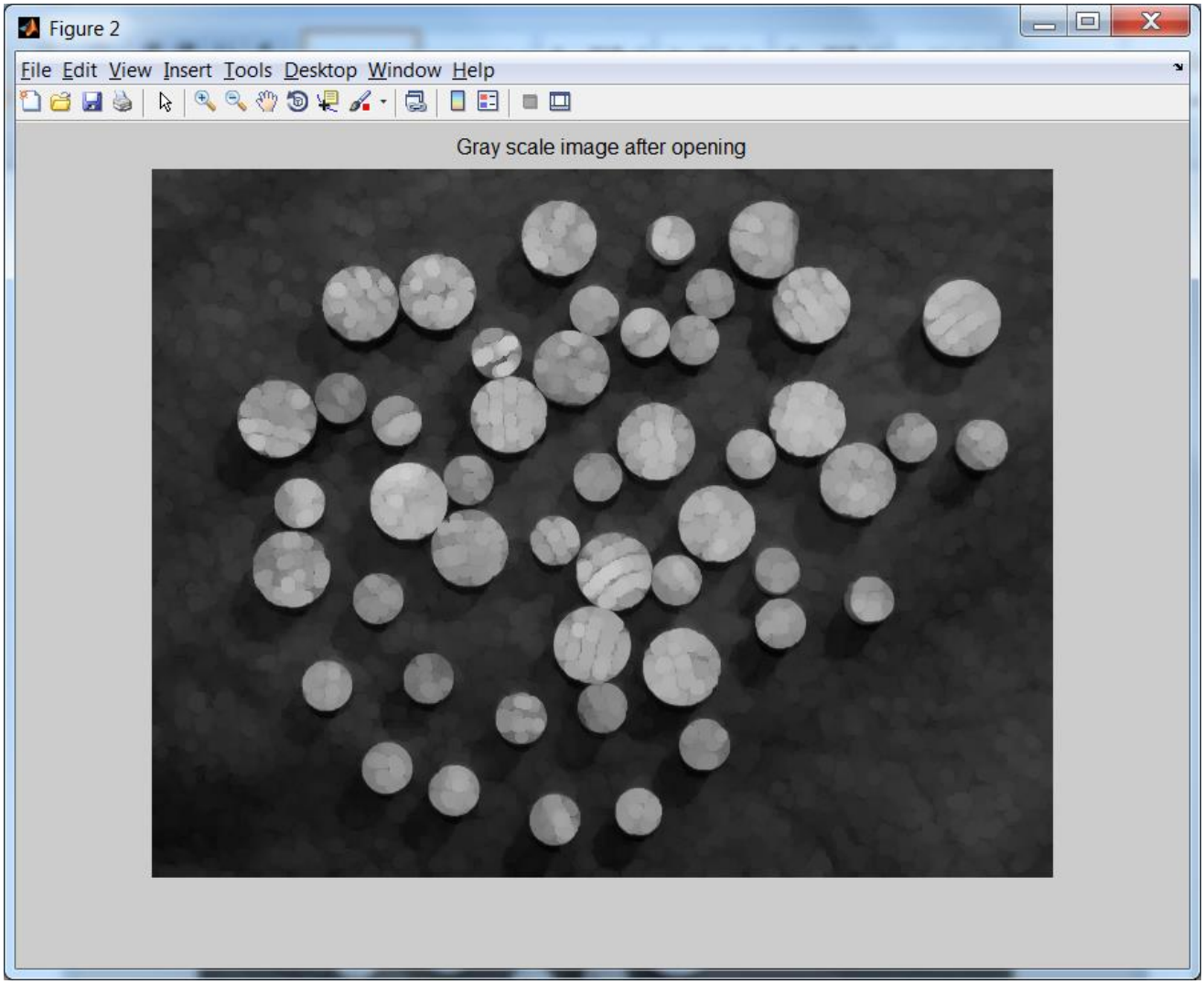


Figure 2
File Edit View Insert Tools Desktop Window Help
Gray scale image after opening

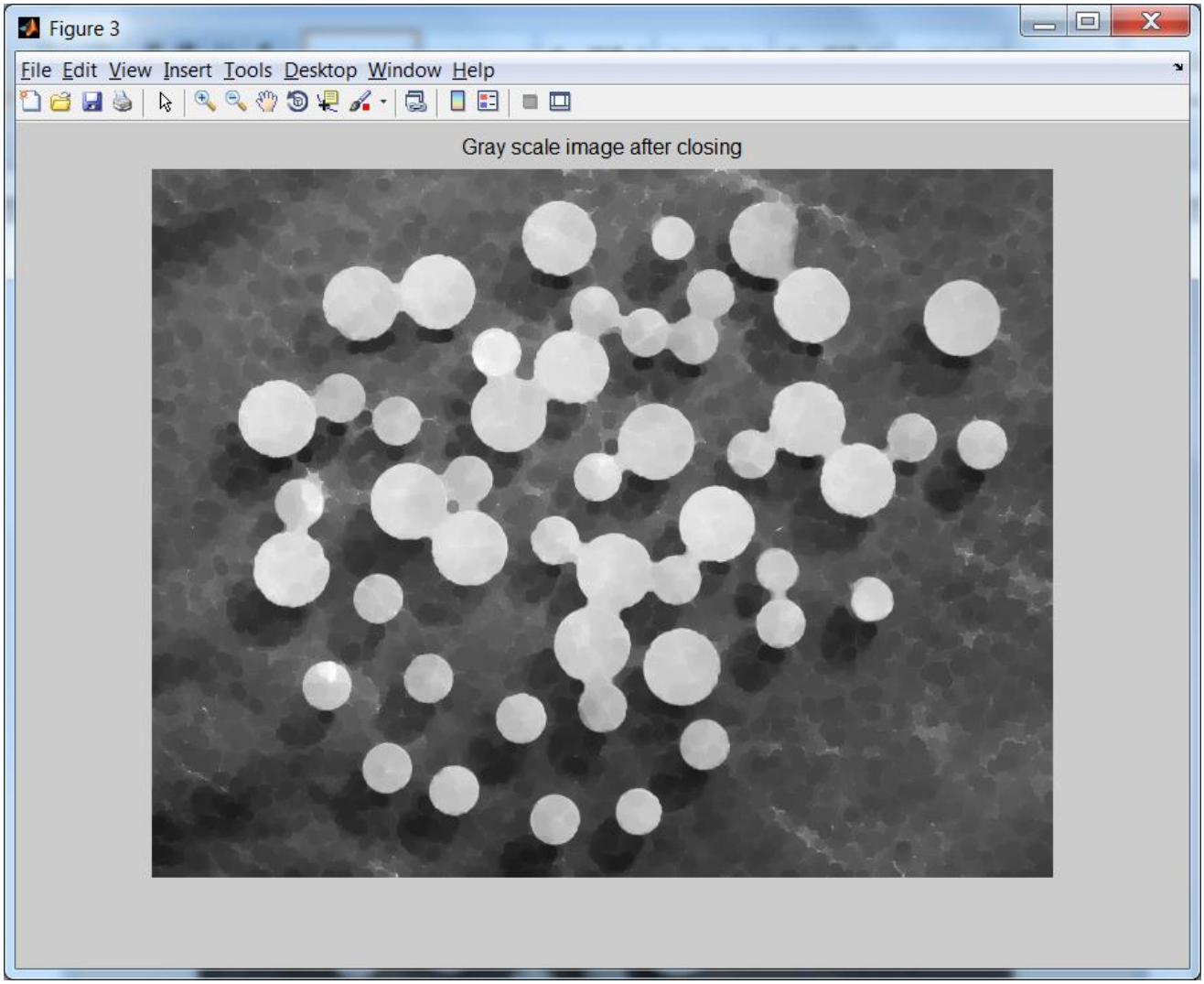


Figure 3
File Edit View Insert Tools Desktop Window Help
Gray scale image after closing