Submitted by – Guneet Singh Mehta

Lab Assignment 2 solutions

Part A

Question 1.

```
function[]=question1()
    s1=3;s2=3;
    delta(1:s1,1:s2)=0;
    delta(1,1)=1;
    D=fft2(delta);
    display(abs(D));
end
% the DFT of an impulse function is a matrix with constant real value like
% a 3X3 Matrix consisting of all 1s
% Reason - All terms in the DFT resolve to Summation of first term of
% impulse matrix(1)X1+0*(exponentials) = 1*impulse(0,0)=1
```

```
ans =

     1     1     1
     1     1     1
     1     1     1
```

Question 2.

```
function[]=question2()
    s1=5;s2=5;
   gaussian=fspecial('gaussian',[s1,s2],0.5);
   display(gaussian);
    F_gaussian=fft2(gaussian);
    F_gaussian_mag=abs(F_gaussian);
    display(F_gaussian_mag);
end
% Obs 1 -The resultant magnitude matrix gives more weight to lower frequency
% coefficients, thus the Gaussian filter has an effect similar to low pass
% filter.
% Obs 2 -The resultant magnitude matrix of Gaussian filter is also symmetric
```

```
gaussian =

    0.0000    0.0000    0.0002    0.0000    0.0000
    0.0000    0.0113    0.0837    0.0113    0.0000
    0.0002    0.0837    0.6187    0.0837    0.0002
    0.0000    0.0113    0.0837    0.0113    0.0000
    0.0000    0.0000    0.0002    0.0000    0.0000


F_gaussian_mag =
```

```
    1.0000    0.8519    0.6145    0.6145    0.8519
    0.8519    0.7258    0.5235    0.5235    0.7258
    0.6145    0.5235    0.3776    0.3776    0.5235
    0.6145    0.5235    0.3776    0.3776    0.5235
    0.8519    0.7258    0.5235    0.5235    0.7258
```

Question 3

```
function[]=question3()
    s1=5;s2=5;
    M=3;N=3;
    h(1:s1,1:s2)=0;
    for i=1:s1
        for j=1:s2
            if(abs(i-floor(s1/2)-1)<M/2&&abs(j-floor(s2/2)-1)<N/2)
                h(i,j)=1;
            end
        end
    end
   display(h);
   H=fft2(h);
   display(abs(H));
end
```

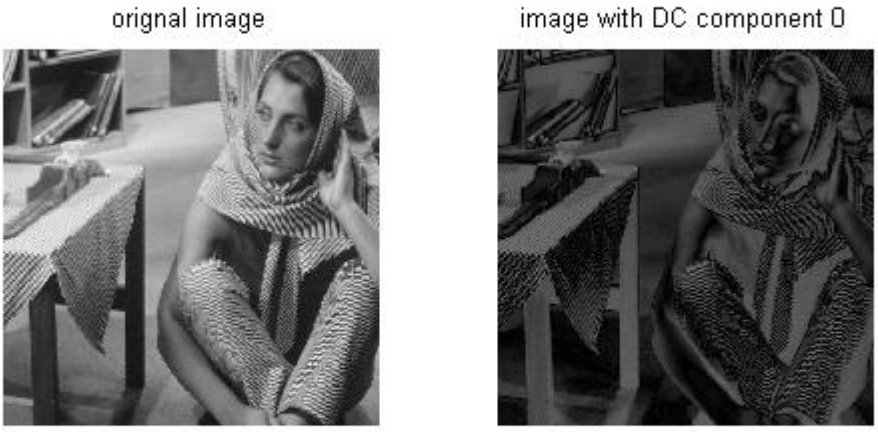
```
% the first coefficient of the DFT of h is the sum of all terms in h
```

```
h =

     0     0     0     0     0
     0     1     1     1     0
     0     1     1     1     0
     0     1     1     1     0
     0     0     0     0     0


ans =

    9.0000    4.8541    1.8541    1.8541    4.8541
    4.8541    2.6180    1.0000    1.0000    2.6180
    1.8541    1.0000    0.3820    0.3820    1.0000
    1.8541    1.0000    0.3820    0.3820    1.0000
    4.8541    2.6180    1.0000    1.0000    2.6180
```

Question 4a.

```
function[]=question4a()
    %Author - Guneet Singh Mehta ,ECE Department, UW Madison
  image=imread('barbara.png');
  if(size(image,3)==3)
     image=rgb2gray(image);
  end
  F=fft2(image);
```

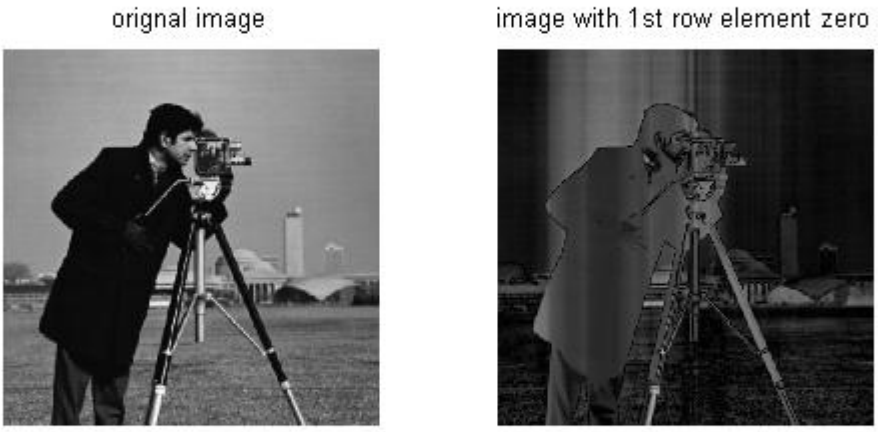
```
  F(1,1)=0;
  imout=uint8(abs(ifft2(F)));
  figure;
  subplot(1,2,1);imshow(image);title('orignal image');
  subplot(1,2,2);imshow(imout);title('image with DC component 0');

end
% It is observed that the resulting image has enhanced edges. Since the DC component F(1,1)
% has been made 0, the DC component of the image has become zero, thus emphasising
% the edges in the image. The absolute value of the resultant image is plotted so that the
%negative values are not clipped
```



Question 4b

```
function[]=question4b()
    %Author - Guneet Singh Mehta ,ECE Department, UW Madison
  image=imread('cameraman.png');
  if(size(image,3)==3)
     image=rgb2gray(image);
  end
  F=fft2(image);
  s1=size(F,1);s2=size(F,2);
  for i=1:s1
      for j=1:s2
          if(i==1)
             F(1,j)=0;
          end
      end
  end
```

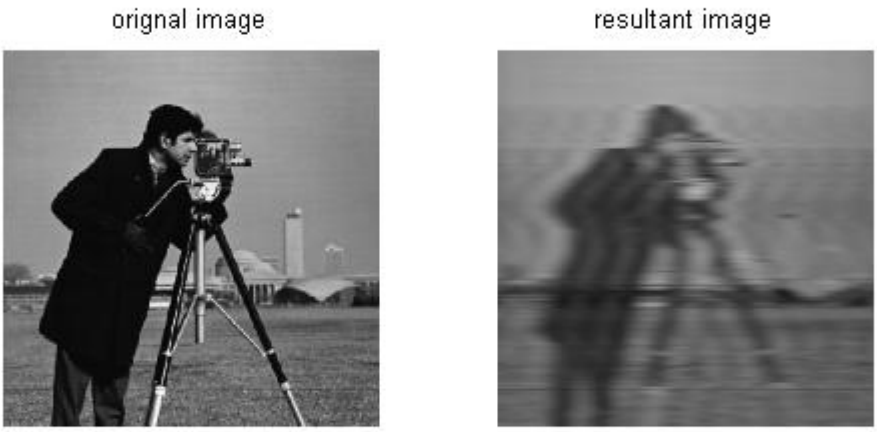
```
  imout=uint8(abs(ifft2(F)));
  %imshow(image,'Border','tight');
  figure;
  subplot(1,2,1);imshow(image,'Border','tight');title('orignal image');
  subplot(1,2,2);imshow(imout,'Border','tight');title('image with 1st row element zero');

end
```



Question 4d

```
function[]=question4d()
    %Author - Guneet Singh Mehta ,ECE Department, UW Madison
 image=imread('cameraman.png');
  if(size(image,3)==3)
     image=rgb2gray(image);
  end
  F=fft2(image);
  s1=size(F,1);s2=size(F,2);
  c=floor(0.02*s2);
  for k1=1:s1
      for k2=1:s2
          if(k2>c)
             F(k1,k2)=0;
          end
      end
  end
  imout=uint8(abs(ifft2(F)));
```

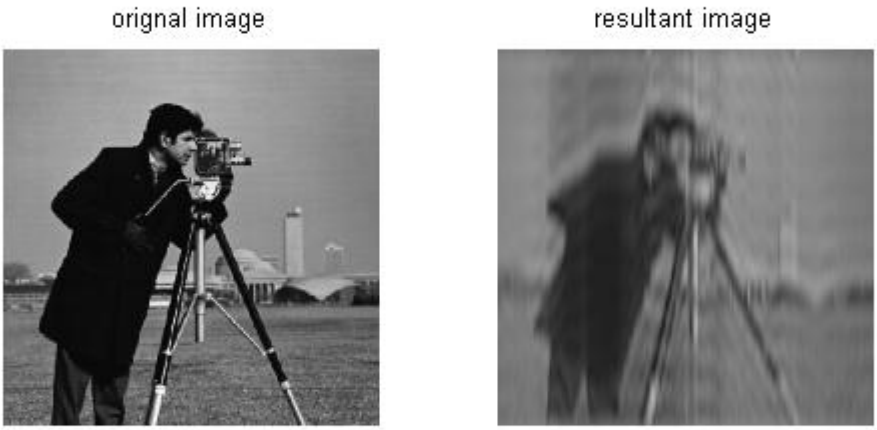
```
  figure;
  subplot(1,2,1);imshow(image);title('orignal image');
  subplot(1,2,2);imshow(imout);title('resultant image');

end
% the resultant image looks as if the camera was moved in the horizontal
% direction while taking the image. The effect can be called as motion
% blurring in horizontal direction
```



Question 4e

```
function[]=question4e()
    %Author - Guneet Singh Mehta ,ECE Department, UW Madison
  image=imread('cameraman.png');
  if(size(image,3)==3)
     image=rgb2gray(image);
  end
  F=fft2(image);
  s1=size(F,1);s2=size(F,2);
  c=floor(0.02*s2);
  for k1=1:s1
      for k2=1:s2
          if(k1>c)
             F(k1,k2)=0;
          end
      end
  end
  imout=uint8(abs(ifft2(F)));
  figure;
```

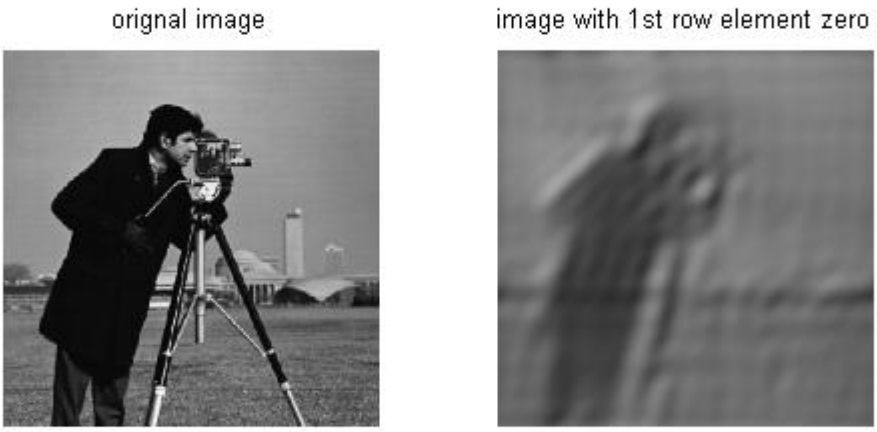
```
    subplot(1,2,1);imshow(image);title('orignal image');
  subplot(1,2,2);imshow(imout);title('resultant image');
end
% the resultant image looks as if the camera was moved in the vertical
% direction while taking the image. The effect can be called as motion
% blurring in vertical direction
```



Question 4f

```
function[]=question4f()
    %Author - Guneet Singh Mehta ,ECE Department, UW Madison
  image=imread('cameraman.png');
  if(size(image,3)==3)
     image=rgb2gray(image);
  end
  F=fft2(image);
  s1=size(F,1);s2=size(F,2);
  c=floor(0.025*s2);
  for k1=1:s1
      for k2=1:s2
          if(k1>c||k2>c)
             F(k1,k2)=0;
          end
      end
  end
  imout=uint8(abs(ifft2(F)));
  figure;
  subplot(1,2,1);imshow(image);title('orignal image');
  subplot(1,2,2);imshow(imout);title('image with 1st row element zero');
```

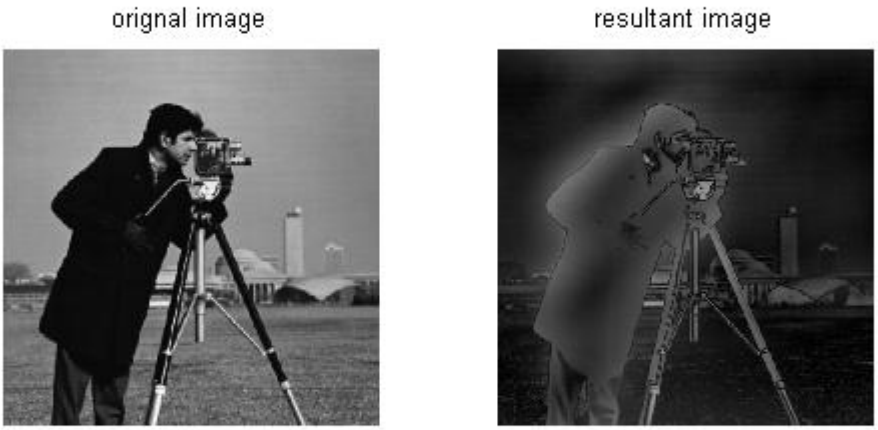
```
end
% This operation leads to motion blurrness associated with camera movement
% in a direction which makes 45degress with the horizontal
```



Question 4g

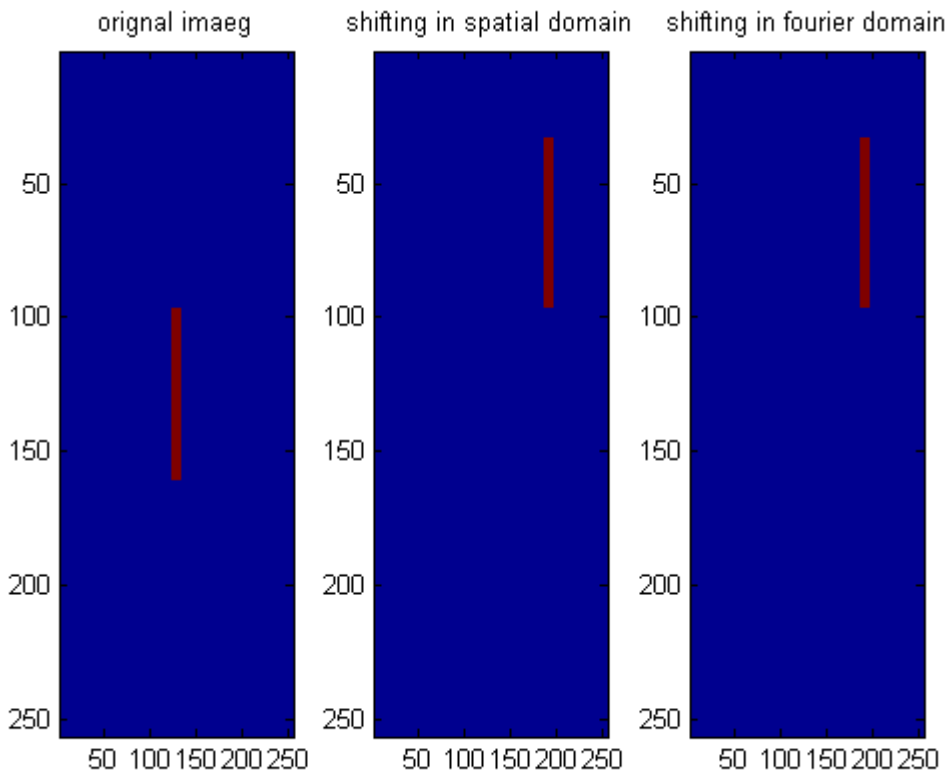
```
function[]=question4g()
    %Author - Guneet Singh Mehta ,ECE Department, UW Madison
  image=imread('cameraman.png');
  if(size(image,3)==3)
     image=rgb2gray(image);
  end
  F=fft2(image);
  s1=size(F,1);s2=size(F,2);
  c=floor(0.01*s2);
  for k1=1:s1
      for k2=1:s2
          if(k1<c&&k2<c)
             F(k1,k2)=0;
          end
      end
  end
  imout=uint8(abs(ifft2(F)));
  figure;
  subplot(1,2,1);imshow(image);title('orignal image');
  subplot(1,2,2);imshow(imout);title('resultant image');

end
```

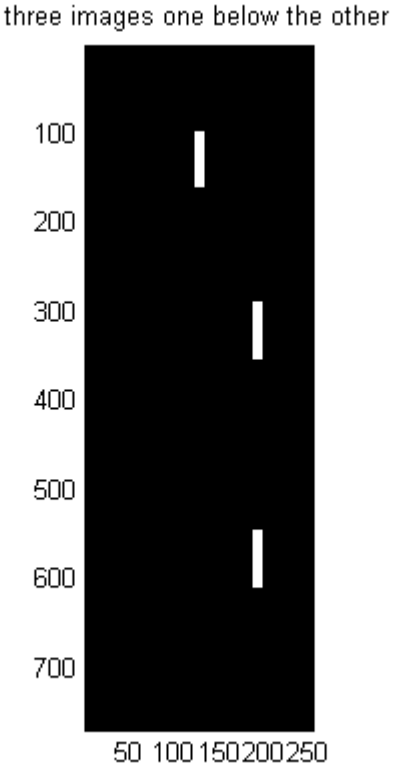
```
% In the resultant image the edges which are not aligned along either x or
% y direction seem to be enhanced
```



Question 5a

```
    M = 256;
    [x,y] = meshgrid(linspace(-2,2,M));
    f1 = rect(y).*rect(5*x);
    f2 = circshift(f1,round([-M/4,M/4]));
    [u,v] = meshgrid(0:(M-1));
 Fx=transform(-0.25,0.25,M,M);
 Fd = fft2(f1).*Fx;
f3 = uint8(real(ifft2(Fd)));
figure;
subplot(1,3,1);imagesc(f1);title('orignal image');
subplot(1,3,2);imagesc(f2);title('shifting in spatial domain');
subplot(1,3,3);imagesc(f3);title('shifting in fourier domain');
figure;imagesc([f1; f2; f3;]);title('three images one below the other');axis image;colormap
gray;
```

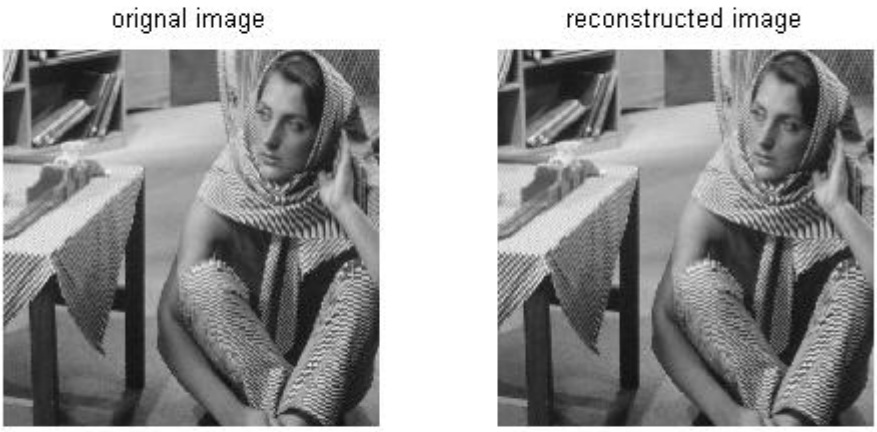

orignal imaeg
shifting in spatial domain
shifting in fourier domain
50
100
150
200
250
50 100 150 200 250


three images one below the other
100
200
300
400
500
600
700
50 100 150 200 250

Question 5b – Doubt –ask

Question 6a

```
function[]=question6a()
    im1=imread('barbara.png');
    if(size(im1,3)==3)
       im1=rgbgray(im1);
    end
   IM1 = fft2(im1);
    mag1 = abs(IM1);
    phase1 = angle(IM1);
    im1_test = uint8(ifft2(mag1.*exp(1i*phase1)));
    figure;
    subplot(1,2,1);imshow(im1);title('orignal image');
    subplot(1,2,2);imshow(im1_test);title('reconstructed image');
end
```



Question 6b

```
function[]=question6b()
    im1=imread('barbara.png');
    if(size(im1,3)==3)
       im1=rgbgray(im1);
    end
   IM1 = fft2(im1);
   s1=size(im1,1);s2=size(im1,2);

   mag1=abs(IM1);
```

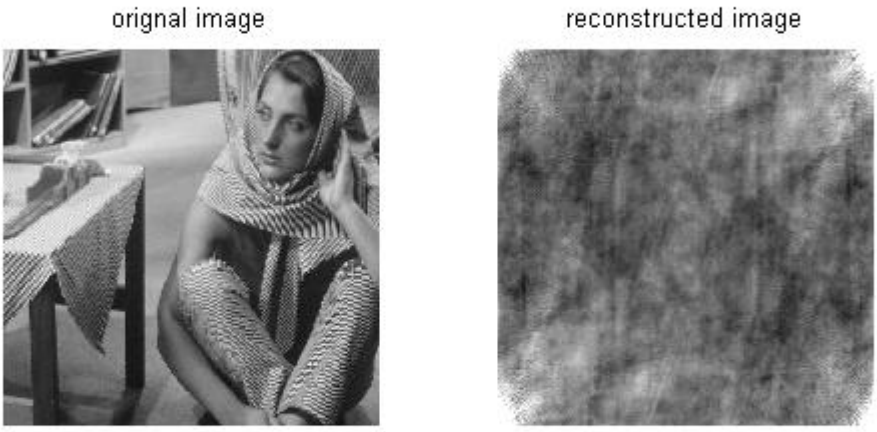
```
    %display(mag1(1:10,1:10));
     mag1 = randi(300,[s1,s2]);
     %display(mag1(1:10,1:10));
     phase1 = angle(IM1);
     im1_test = real(ifft2(mag1.*exp(1i*phase1)));
      figure;
     subplot(1,2,1);imshow(im1);title('orignal image');
     subplot(1,2,2);imshow(im1_test);title('reconstructed image');

end
```

```
% Retaining the phase of the image, results in the some similarities being shown in the
% reconstructed image. For example ,the outline of the woman's face can be seen in the
% reconstructed image
```

Question 6c

```
function[]=question6c()
    im1=imread('barbara.png');
    if(size(im1,3)==3)
       im1=rgbgray(im1);
    end
   IM1 = fft2(im1);
   s1=size(im1,1);s2=size(im1,2);

   mag1=abs(IM1);
    phase1 = angle(IM1);
   % display(phase1(1:10,1:10));
    phase1=randn(1:s1,1:s2);
```

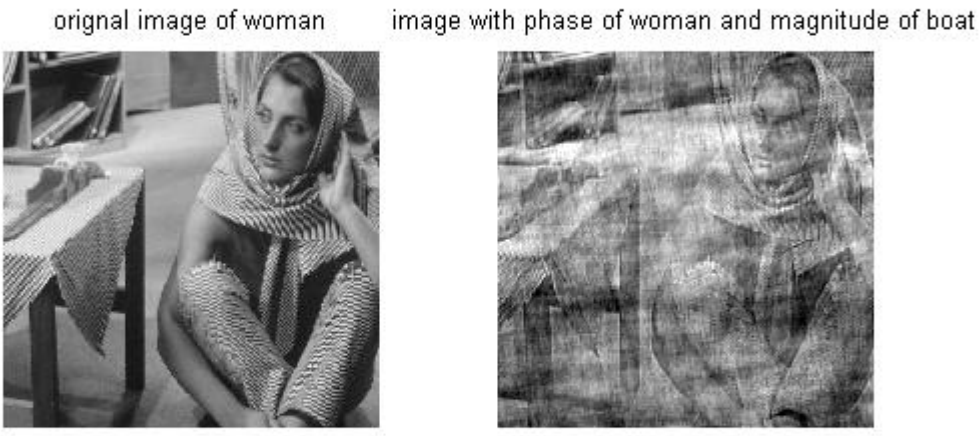
```
    im1_test = uint8(real(ifft2(mag1.*exp(1i*phase1))));
     figure;
    subplot(1,2,1);imshow(im1);title('orignal image');
    subplot(1,2,2);imshow(im1_test);title('reconstructed image');
end
%The reconstructed image has no similarities with the orignal image. The
%reconstructed image appears to be a random smudge. The phase information
%is required so that the reconstructed image has some similarity with the
%orignal image. While the magnitude infomation is also necessary , but
% phase information is more important for visual similarity
```



Question 6d

```
function[]=question6d()
    im1=imread('barbara.png');
    im2=imread('boat.png');
    if(size(im1,3)==3)
       im1=rgbgray(im1);
    end
    if(size(im2,3)==3)
       im2=rgbgray(im2);
    end
    %figure;imshow(im1);
    %figure;imshow(im2);
    F1=fft2(im1);
    F2=fft2(im2);

    F1_abs=abs(F1);
    F2_abs=abs(F2);
```

```
    F1_ang=angle(F1);
    F2_ang=angle(F2);

    F1_new=F1_abs.*exp(1i*F2_ang);
    F2_new=F2_abs.*exp(1i*F1_ang);

    imout1=uint8(ifft2(F1_new));
    imout2=uint8(ifft2(F2_new));

    figure;
    subplot(1,2,1);imshow(im1);title('orignal image of woman')
    subplot(1,2,2);imshow(imout2);title('image with phase of woman and magnitude of boat');

    figure;
    subplot(1,2,1);imshow(im2);title('orignal boat image');
    subplot(1,2,2);imshow(imout1);title('image with phase of boat and magnitude of woman');

end
% the reconstructed images have similarities to the image whose phase is
% used to construct the image. That is, the spatial information leading to
% similarity in perception is carried by the phase information and not
% magnitude
```



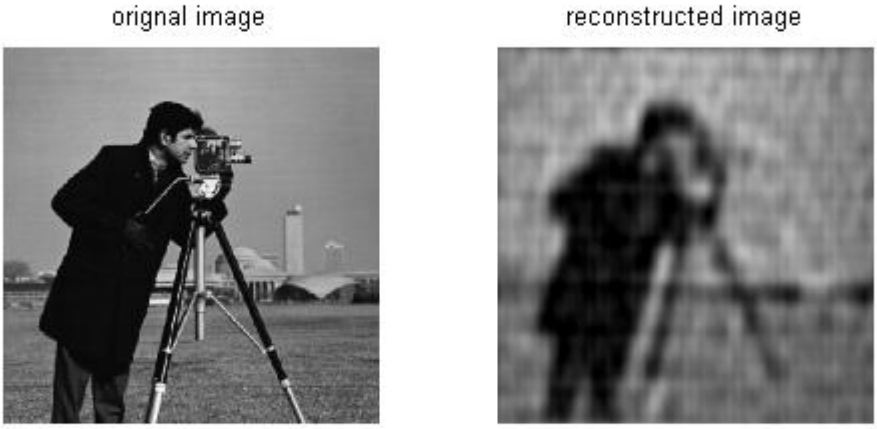
orignal boat image

image with phase of boat and magnitude of woman

Question 7a

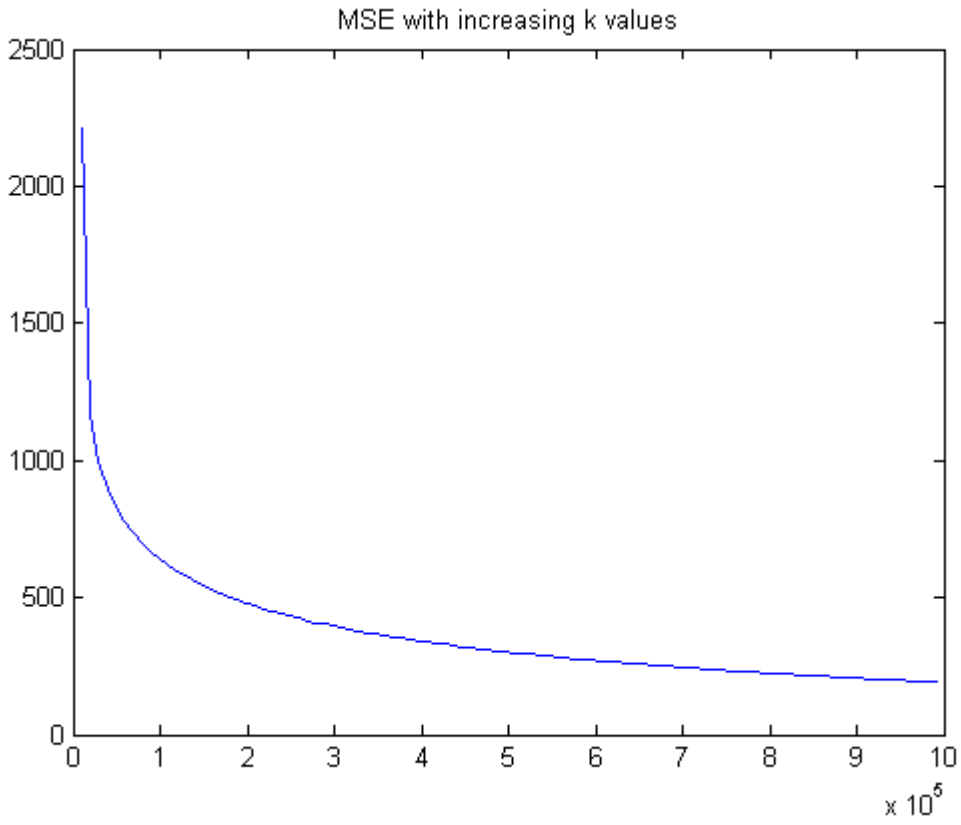
```
function[]=question7a2()
    image=imread('cameraman.png');k=1000;
    % using the k highest coefficients  in DFT
    show_figs=1;
    F=fft2(image);
    [s1,s2]=size(image);
    temp=k;
    mag=abs(F);phase=angle(F);
    a(1:s1*s2)=0;

    mags=sort(mag(:),'descend');
    k_threshold=mags(k+1);
    mag2=mag;
    for i=1:s1
        for j=1:s2
            if(mag2(i,j)<k_threshold)
               mag2(i,j)=0;
            end
        end
    end
    F2=mag2.*exp(1i*phase);imout=uint8(ifft2(F2));
    if(show_figs==1)
        figure;
        subplot(1,2,1);imshow(image);title('orignal image');
        subplot(1,2,2);imshow(imout);title('reconstructed image');
```

```
    end
end
```



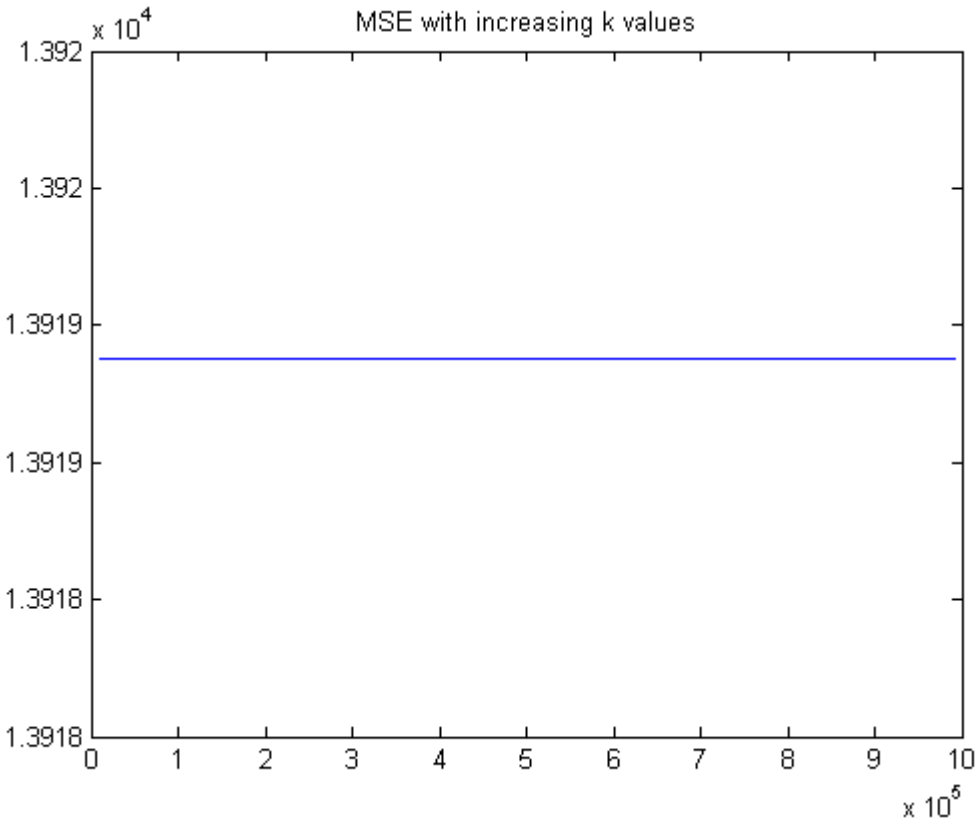
Question 7b

```
function[]=question7b()
    image=double(imread('cameraman.png'));
    [s1,s2]=size(image);
    counter=1;
    steps=100;
    MSE(1:steps)=0;
    indices=1:steps;
    for k=10:100:100*steps
       imout=question7a(image,k);
       MSE(counter)=0;
       imdiff=image-imout;
       MSE(counter)=sum(imdiff(:).*imdiff(:));
%        for i=1:s1
%            for j=1:s2
%                MSE(counter)=MSE(counter)+(image(i,j)-imout(i,j))^2;
%            end
%        end
       MSE(counter)=MSE(counter)/(s1*s2);
       counter=counter+1;
       fprintf('%d ',counter);
    end
    indices=indices*k;
    figure;plot(indices,MSE);title('MSE with increasing k values');
end
```



Question 7c

```
function[]=question7c()
    image=double(imread('cameraman.png'));
    im_noise=imnoise(image,'gaussian',0,0.05);
    [s1,s2]=size(image);
    counter=1;
    steps=100;
    MSE(1:steps)=0;
    indices=1:steps;
    for k=10:100:100*steps
       imout=question7a(im_noise,k);
       MSE(counter)=0;
       imdiff=image-imout;
       MSE(counter)=sum(imdiff(:).*imdiff(:));
%        for i=1:s1
%            for j=1:s2
%                MSE(counter)=MSE(counter)+(image(i,j)-imout(i,j))^2;
%            end
%        end
       MSE(counter)=MSE(counter)/(s1*s2);
       counter=counter+1;
       fprintf('%d ',counter);
    end
    indices=indices*k;
    figure;plot(indices,MSE);title('MSE with increasing k values');
end
```



Question7d

Traversing the DFT in the manner shown below and retaining only the k coefficients that come first.

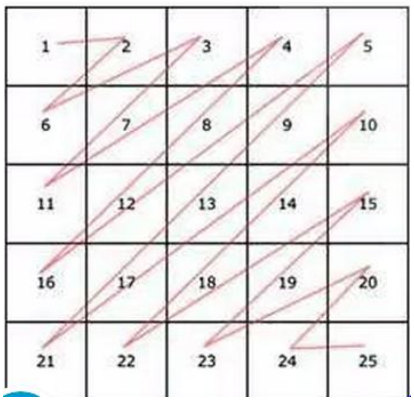


Figure 1 Image Courtsey sumeetkataria.com - http://www.sumeetkataria.com/2012/12/print-two-dimensional-matrix-in-zigzag-form/

```
function[]=question7d()
    image=imread('cameraman.png');
    if(size(image,3)==3)
       image=double(rgb2gray(image));
    end
    [s1,s2]=size(image);
    counter=1;
    steps=100;
    MSE(1:steps)=0;
```

```
    indices=10:100:100*steps;
    for k=10:100:100*steps
       [imout,k_actual]=question7d_sub(image,k);
       MSE(counter)=0;
       imdiff=image-imout;
       MSE(counter)=sum(imdiff(:).*imdiff(:));
%        for i=1:s1
%            for j=1:s2
%                MSE(counter)=MSE(counter)+(image(i,j)-imout(i,j))^2;
%            end
%        end
       MSE(counter)=MSE(counter)/(s1*s2);
       indices(counter)=k_actual;
       counter=counter+1;
     %  fprintf('%d ',counter);
    end
    %display(indices);
    figure;plot(indices,MSE);title('MSE with increasing k values');
end

function[imout,k_actual]=question7d_sub(image,k)
    % using the k highest coefficients  in DFT
    show_figs=0;
    F=fft2(image);
    [s1,s2]=size(image);
    mag=abs(F);phase=angle(F);
    mag2(1:s1,1:s2)=0;

    k_temp=k;sum=1;
    while(k_temp-sum>=0)
       for i=1:sum
           j=sum-i+1;
           mag2(i,j)=mag(i,j);
       end
       k_temp=k_temp-sum;
       sum=sum+1;
    end
    k_actual=k-(k_temp);
%    % display(size(mag2));
%     display(size(phase));%pause(5);
    F2=mag2.*exp(1i*phase);imout=uint8(ifft2(F2));
    if(show_figs==1)
        figure;
        subplot(1,2,1);imshow(image);title('orignal image');
        subplot(1,2,2);imshow(imout);title('reconstructed image');
    end
    imout=double(imout);
end
```

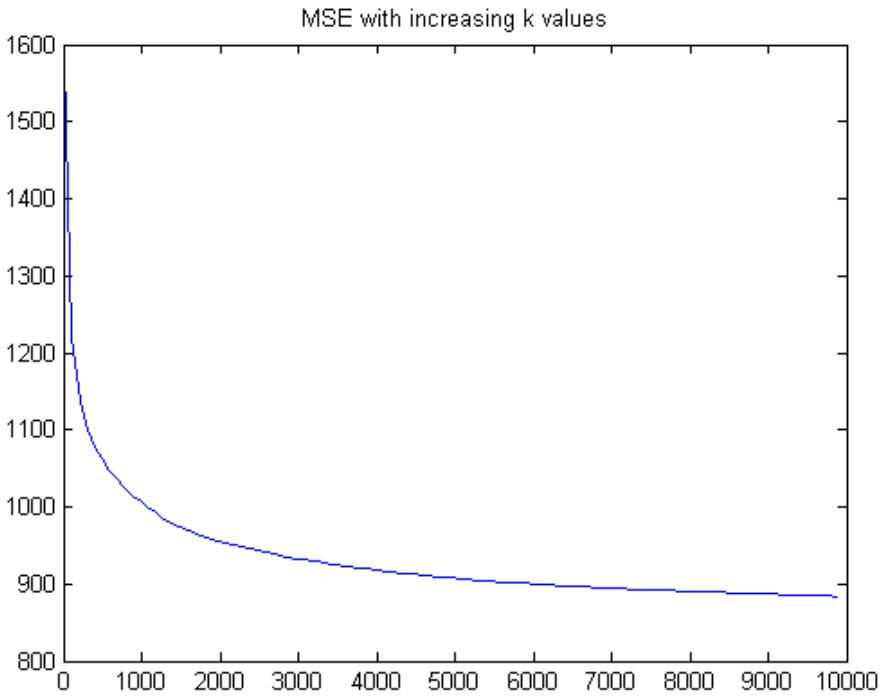


Figure 2 Question 7d

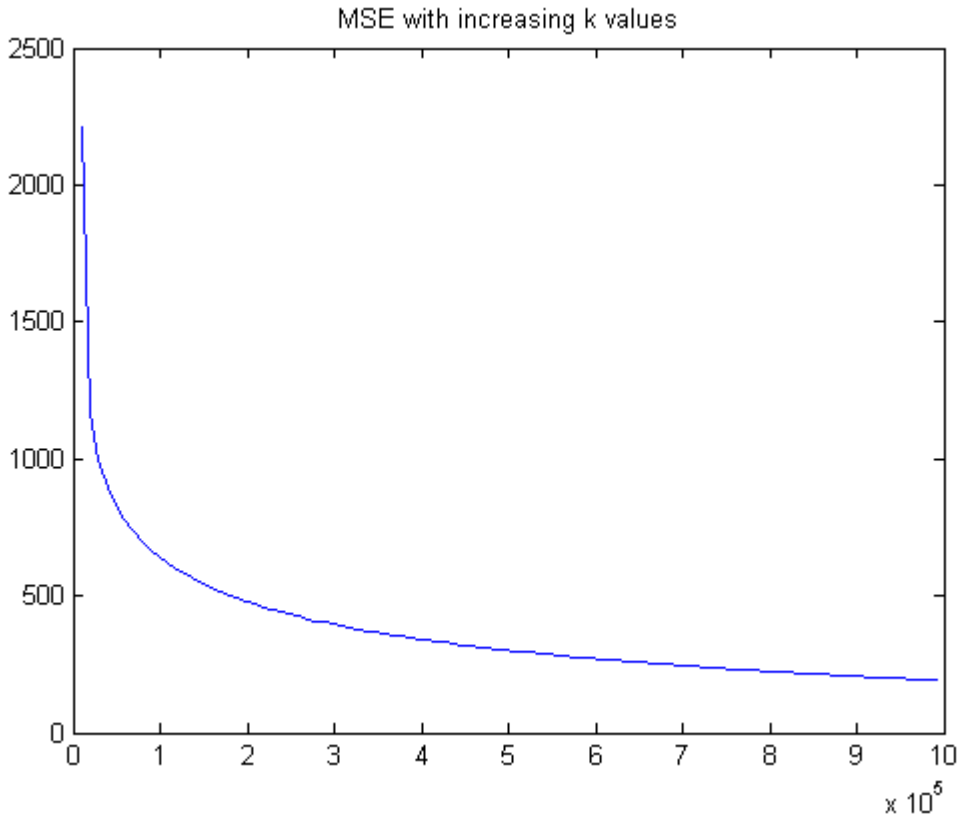


Figure 3 Question 7b MSE vs k

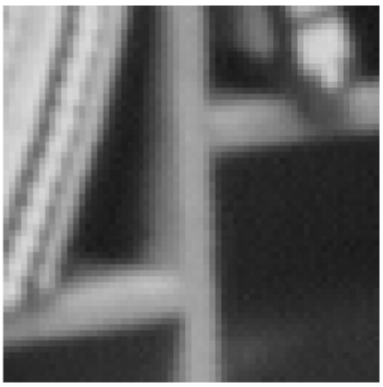
The MSE values for modified algorithm is more than the maximum k coefficients in DFT. Thus the proposed algorithm does not fair better than the maximum k coefficients algorithm

Question 8

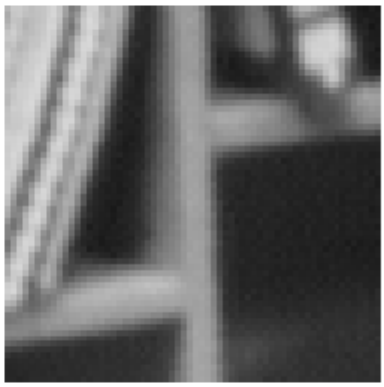
```
function[]=Bquestion8()
    image=imread('barbara.png');
    if(size(image,3)==3)
       image=rgb2gray(image);
    end
    image=image(1:64,1:64);
    %image=[1 2 ;3 4 ];
    [s1,s2]=size(image);
%     if(2^floor(log2(s1))~=s1||2^floor(log2(s2))~=s2)
%        display('program applicable only for images with dimensions with the power of 2');
%        display(size(image));
%        return;
%     end
    for i=1:s1
        F(i,:)=questionB8_sub(image(i,:));
        %display(image(i,:));
    end
    %display(F);
    for j=1:s2
        %display(transpose(image(:,j)));
        F2(:,j)=transpose(questionB8_sub(transpose(F(:,j))));
    end
    F_base(1:s1,1:s2)=fft2(image);

    figure;
    subplot(1,2,1);imshow(image);title('orignal image');
    subplot(1,2,2);imshow(uint8(abs(ifft2(F2))));title('Reconstructed image from computed
FFT');

    figure;
    subplot(1,2,1);imagesc(log(1+abs(fftshift(F_base))));title('orignal Fourier Tranform');
    subplot(1,2,2);imagesc(log(1+abs(fftshift(F2))));title('Computed Fourier Transform');
end

function [F_comb]=questionB8_sub(image)
    %image=[1 1 1 1 1 1 1 1];
    num_steps=log2(size(image,2));
%     if(2^floor(log2(num_steps))~=num_steps)
%        display('program only for power of 2');
%         return;
%     end
    %display(num_steps);
    test=[1 1];
    %fft1d(test,0);
    F_comb=fft1d_g(image);
   % display(F_comb);
end

function[F_comb]=fft1d_g(seq)
```

```
    num_entries=size(seq,2);
    F_even(1:num_entries/2)=0;
    F_odd(1:num_entries/2)=0;
    F_comb(1:num_entries)=0;
    seq_even(1:num_entries/2)=0;
    seq_odd(1:num_entries/2)=0;

    for i=1:num_entries/2
       seq_even(i)=seq(2*i-1);
       seq_odd(i)=seq(2*i);
    end
    if(num_entries==2)
       F_comb=[seq(1)+seq(2),seq(1)-seq(2)];
        return;
    else
       for u=1:num_entries
%            clc;
%            display(u);
          F_comb(u)=fft1d(seq_even,u-1)+exp(-2*pi*1i*(u-1)/num_entries)*fft1d(seq_odd,u-1);
       end
    end

end

function[result]=fft1d(seq,u)

    num_entries=size(seq,2);
%      display(num_entries);
     if(num_entries==1)
        result=seq;

     else
        seq_even(1:num_entries/2)=0;
        seq_odd(1:num_entries/2)=0;
        for i=1:num_entries/2
           seq_even(i)=seq(2*i-1);
           seq_odd(i)=seq(2*i);
        end
%         display(seq_even);
%         display(seq_odd);
        %pause(3);
        result=fft1d(seq_even,u)+exp(-2*pi*1i*u/num_entries)*fft1d(seq_odd,u);
     end
end
```

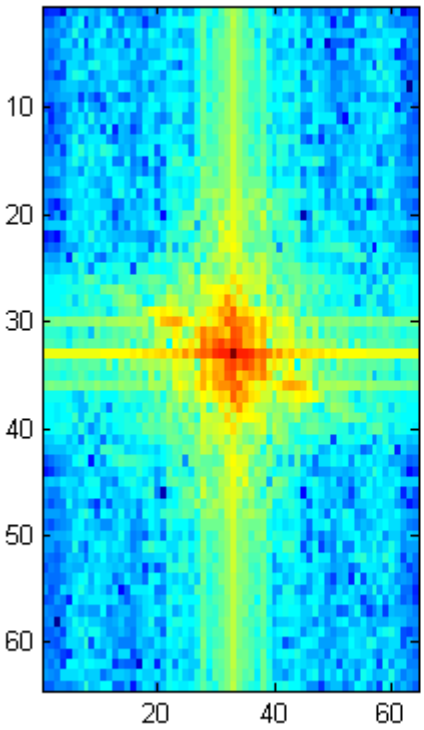
orignal image



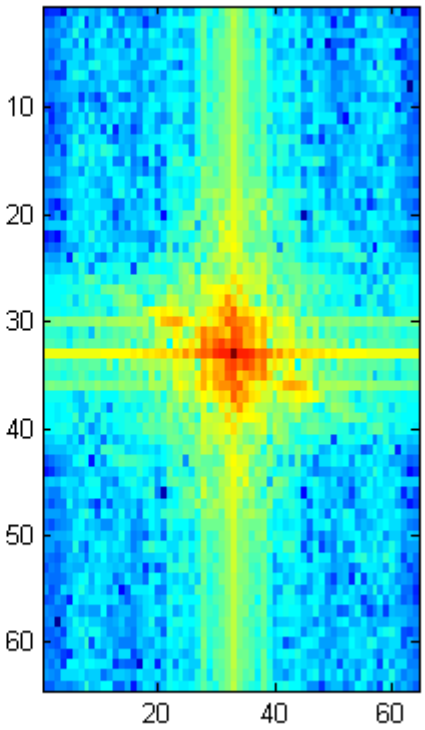
Reconstructed image from computed FFT



orignal Fourier Tranform



Computed Fourier Transform



Part B

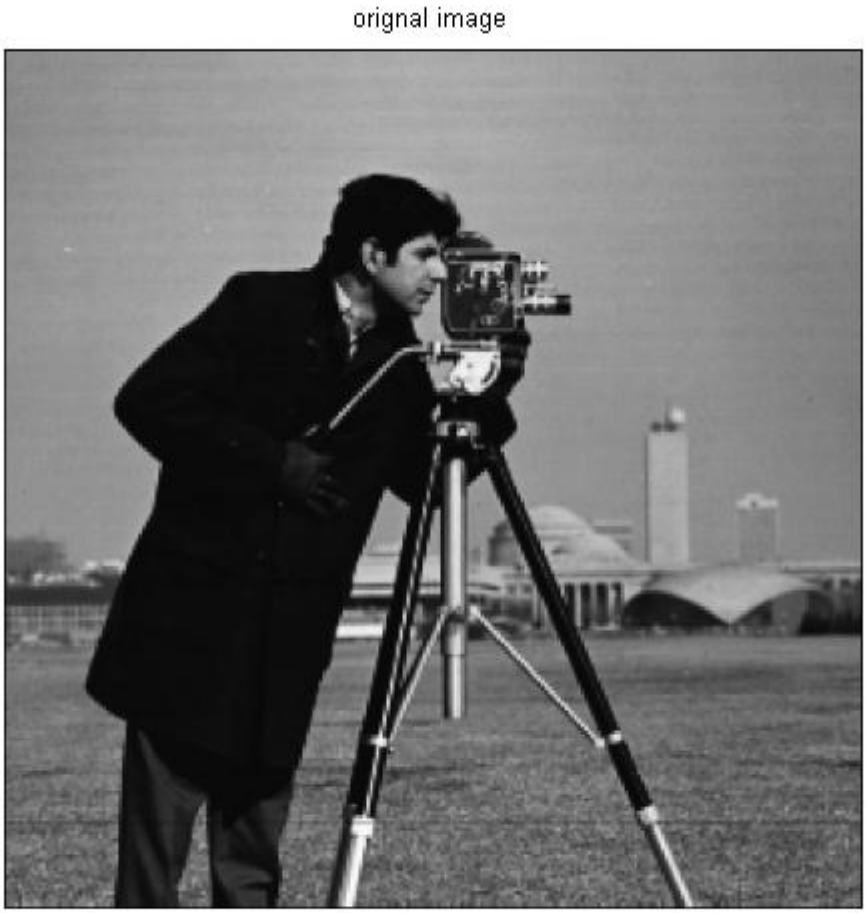
Question B1 –

```
function []=Bquestion1()
% filtering image with gaussian function
% use testimage
         image=imread('cameraman.png');
         if(size(image,3)==3)
            image=rgb2gray(image);
         end
         [s1,s2]=size(image);
         h1=fspecial('gaussian');
         h1=padarray(h1,[floor((s1-size(h1,1))),floor((s2-size(h1,2)))],'post');
         %H1=fftshift(fft2(h1));

         F=fft2(image);
        H=fft2(h1);
        Y=(F).*(H);
        figure;imshow(image);title('orignal image');
        figure;

subplot(1,2,1);colormap(jet);imagesc(abs(1+log(abs(fftshift(H)))));colorbar;title('Gaussian
Filter in Freq Domain');
        subplot(1,2,2);imshow(uint8(abs(ifft2(Y))));title('Gaussian filtered Image');
        %pause(5);

        h2=rect_fn;
         H2=fftshift(fft2(h2));

        F=fft2(image);
        H=fft2(h2);
        Y=(F).*(H);
        figure;

subplot(1,2,1);colormap(jet);imagesc(abs(1+log(abs(fftshift(H)))),[0,2]);colorbar;title('Rect
function in Freq Domain');
        subplot(1,2,2);imshow(uint8(ifft2(Y)));title('Image filtered by Rect function');

     function[h]=rect_fn()
         a1=5;a2=5;
        for i=1:a1
            for j=1:a1
                h(i,j)=1/(a1*a2);
            end
        end
        h=padarray(h,[floor((s1-size(h,1))),floor((s2-size(h,2)))],'post');
        %h=padarray(h,[floor((s1-size(h,1))),floor((s2-size(h,2)))],'post');
        display(size(h));
    end

end
%The Gaussian Filter keeps the low frequency component as it is while
%attenuating High frequency components
%The averaging filter has a similar effect in the mid region of the
%frequency response. The low freq components are kept as it is, and as we
```

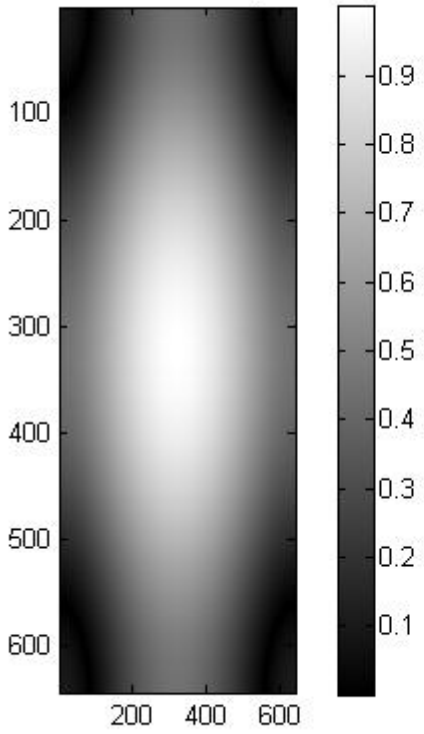
```
% distance from the center increases the attenuation of Freq increases. But
% after a certain distance the coefficient increases again but since most
% of the frequency information is contained near the center, the overall
% effect is that of low pass filter
```

```
Warning: Image is too big to fit on screen; displaying at 67%
```

```
ans =

   643   643
```



Gaussian Filter in Freq Domain



Gaussian filtered Image

Rect function in Freq Domain

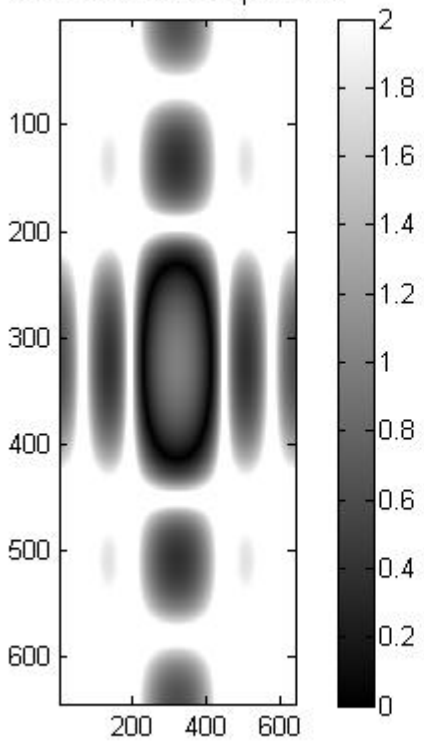
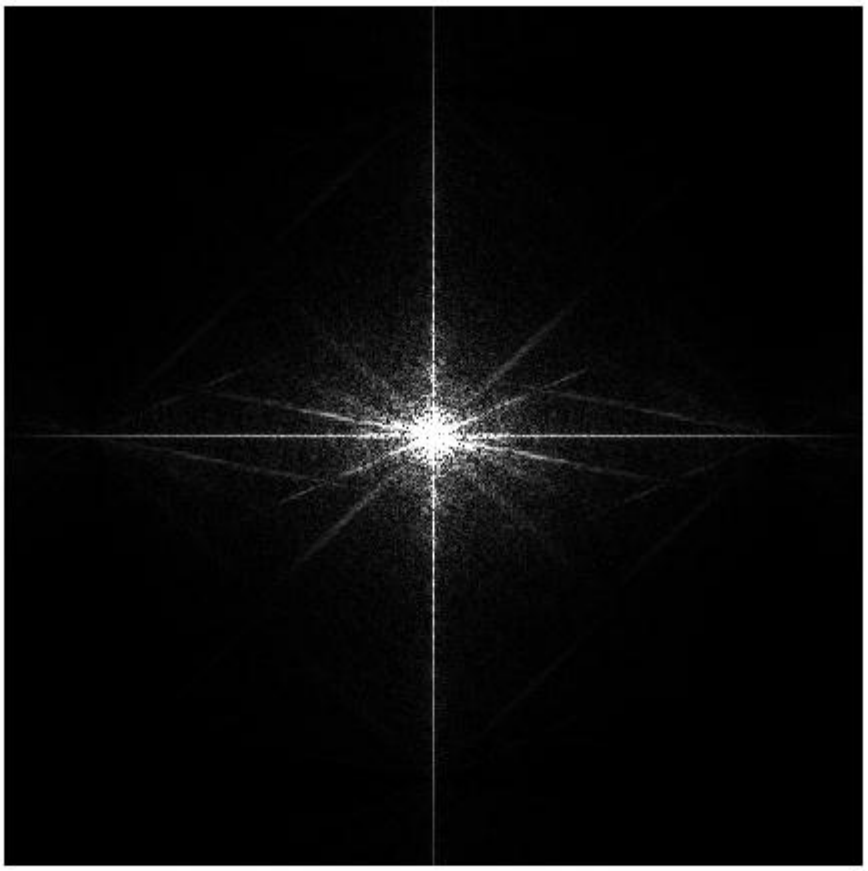


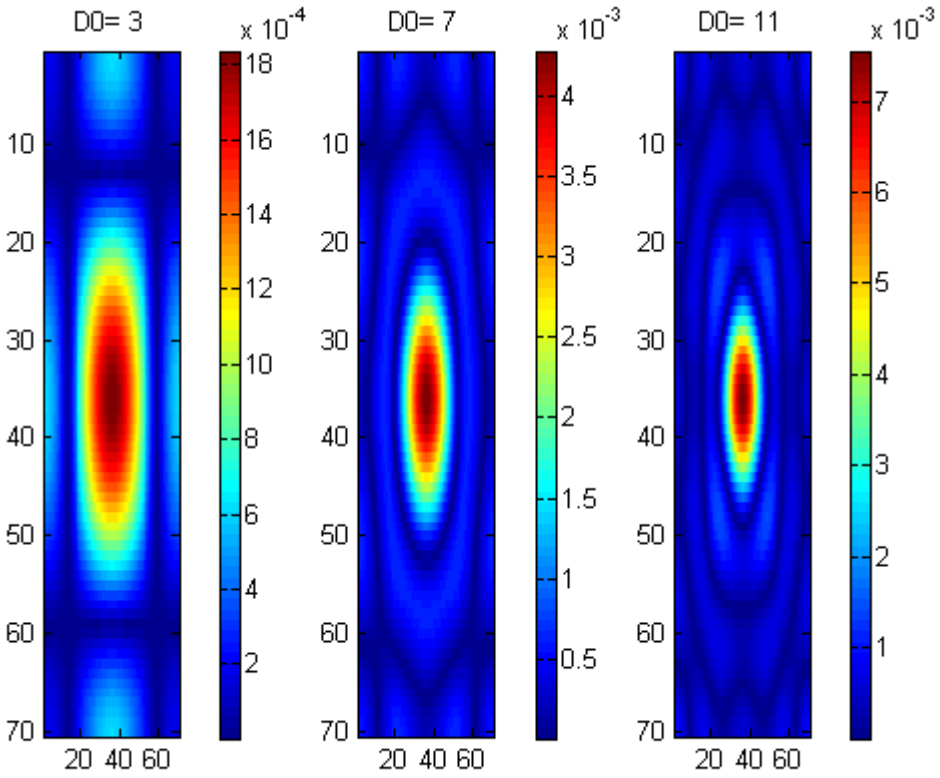
Image filtered by Rect function

Question B2

```
function []=Bquestion2()
    image=imread('cameraman.png');
    if(size(image,3)==3)
       image=rgb2gray(image);
    end

   %code goes here
   F=fft2(image);
   F=fftshift(F);
     [s1,s2]=size(image);
     A=abs(F);
     max1=max(A(:));
%     display(max1);
     A2=A/max1*255;
     %display(max(A2(:)));
     figure;imshow(A2);

     D0=20;
     for i=1:s1
         for j=1:s2
            if((i-s1/2)^2+(j-s2/2)^2>D0)
               F(i,j)=0 ;
            end
         end
     end
     figure;
     s1=70;s2=70;
     k=1;steps=3;
     for D0=3:4:3+4*(steps-1)
         F3(1:s1,1:s2)=1;
        for i=1:s1
            for j=1:s2
               if((i-s1/2)^2+(j-s2/2)^2>D0)
                    F3(i,j)=0 ;
               end
            end
        end
        f3=fftshift(abs(ifft2((F3))));
        str=['D0= ' num2str(D0)];
        subplot(1,steps,k);imagesc(f3);colorbar;title(str);
        k=k+1;

     end

   F2=ifftshift(F);
   imout=uint8(ifft2(F2));

    figure;
    subplot(1,2,1);imshow(image);title('orignal image');
    subplot(1,2,2);imshow(imout);title('Low pass image,with D0=20');

end
% for different values of D0 the level of blurring changes. as the value of
% D0 decreases the cuttoff frequency of the low pass filter decreases, thus
```

```
% increasing the blurrness
% As the filter low pass freq increases, the weightage given to the central
% pixel decreases, the weights of the central pixel is increasing, reducing
% the effect of pixels far off from the current pixel , thus reducing
% blurring
```





orignal image

Low pass image,with D0=20

Question B3

```
function []=Bquestion3()
    image=imread('cameraman.png');
    if(size(image,3)==3)
       image=rgb2gray(image);
    end

    [s1,s2]=size(image);
    border=floor(s1*0.3);
    im1(1:s1+border,1:s2+border)=uint8(0);
    im2(1:s1+border,1:s2+border)=uint8(0);
    im1(1:s1,1:s2)=image;
    im2(border/2:s1+border/2-1,border/2:s2+border/2-1)=image;
    figure;
    subplot(2,1,1);imshow(im1);
    subplot(2,1,2);imshow(im2);
    imout1=low_pass(im1,200);
    imout2=low_pass(im2,200);
end

function[imout]=low_pass(image,D0)

   %code goes here
   F=fft2(image);
   F=fftshift(F);
     [s1,s2]=size(image);
     A=abs(F);
     max1=max(A(:));
     %display(max1);
     A2=A/max1*255;
     %display(max(A2(:)));
     %figure;imshow(A2);


     for i=1:s1
         for j=1:s2
            if((i-s1/2)^2+(j-s2/2)^2>D0)
               F(i,j)=0 ;
            end
         end
     end
   F2=ifftshift(F);
   imout=uint8(abs(ifft2(F2)));

    figure;
    subplot(1,2,1);imshow(image);title('orignal image');
    subplot(1,2,2);imshow(imout);title('Transformed image');

end
% The image with uniform border accross it is preferred because after
% filtering the resulting image can be cropped back to get the resultant
% image of the same size as the orignal one
% In contrast if the image is padded only on one side, then the borders on
% the pixel intensities on non padded sides of the image are lost
```

orignal image

Transformed image

orignal image

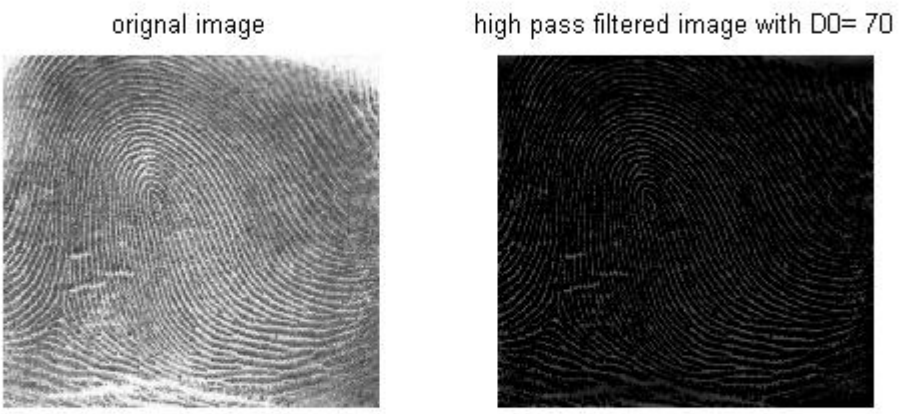
Transformed image

Question B4

```
function []=Bquestion4()
    image=imread('ThumbPrint.tif');
    if(size(image,3)==3)
       image=rgb2gray(image);
    end
    imout=high_pass(image,70);


    function[imout]=high_pass(image,D0)

       %code goes here
       F=fft2(image);
       F=fftshift(F);
         [s1,s2]=size(image);
         A=abs(F);
         max1=max(A(:));
         display(max1);
         A2=A/max1*255;
         display(max(A2(:)));
         %figure;imshow(A2);


         for i=1:s1
             for j=1:s2
                if((i-s1/2)^2+(j-s2/2)^2<D0)
                   F(i,j)=0 ;
                end
```

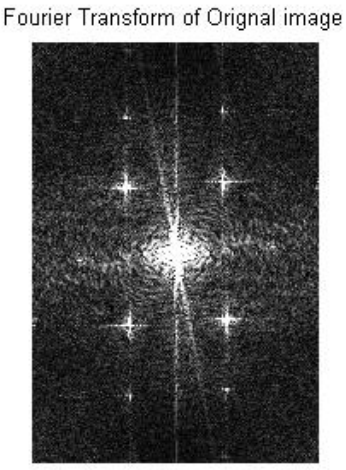
```
            end
         end
       F2=ifftshift(F);
       imout=uint8(ifft2(F2));
        str=['high pass filtered image with D0= ' num2str(D0)];
        figure;
        subplot(1,2,1);imshow(image);title('orignal image');
        subplot(1,2,2);imshow(imout);title(str);

    end
end
```

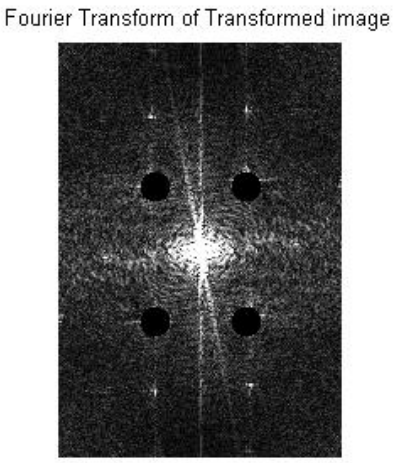
```
Warning: TIFF library error - '_TIFFVSetField:  G:\Acads -UW Madison\First
Sem\Image Processing\Labs\ImgProcessing\Lab2\ThumbPrint.tif: Null count for "Tag
34022" (type 1, writecount -3, passcount 1).' - file may be corrupt.

max1 =

   140874449


ans =

   255

Warning: Displaying real part of complex input.
```

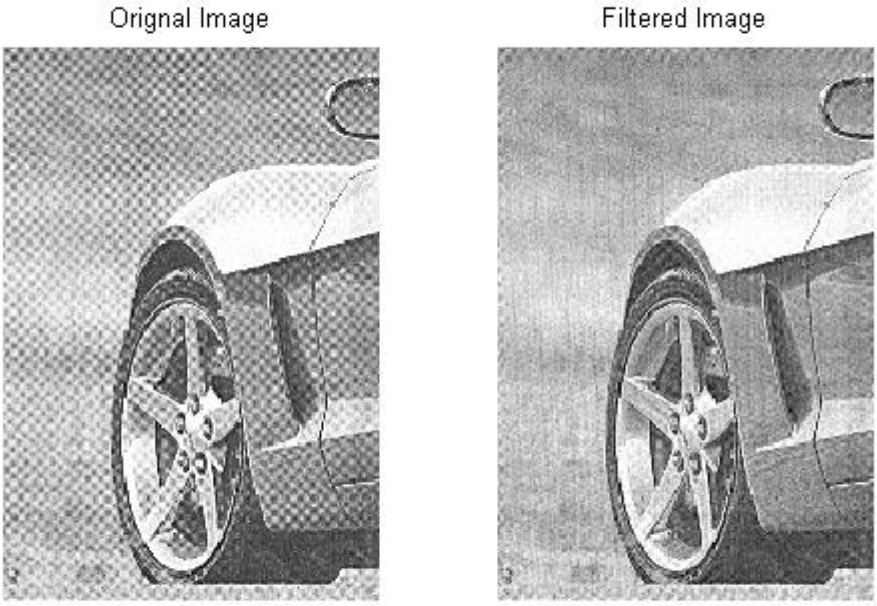



```
function []=Bquestion5()
    image=imread('MoirePattern.tif');
```

```
    if(size(image,3)==3)
       image=rgb2gray(image);
    end
    F=fftshift(fft2(image));
    show_F(F,'Fourier Transform of Orignal image');
       Fcopy=F;
       r=80;
       Fcopy=notch_filter(Fcopy,166,58,r);
       Fcopy=notch_filter(Fcopy,86,58,r);
       Fcopy=notch_filter(Fcopy,86,112,r);
       Fcopy=notch_filter(Fcopy,166,112,r);
    show_F(Fcopy,'Fourier Transform of Transformed image');
    y=uint8(abs(ifft2(Fcopy)));
    figure;
    subplot(1,2,1);imshow(image);title('Orignal Image');
    subplot(1,2,2);imshow(y);title('Filtered Image');
end

function[]=show_F(F,str)
         A=abs(F);
         max1=max(A(:));
         A2=A/max1*255;
         figure;imshow(A2);title(str);
end

function[F]=notch_filter(F,x,y,r)
    [s1,s2]=size(F);
    for i=1:s1
        for j=1:s2
            D=(i-x)^2+(j-y)^2;
            if(D<r)
               F(i,j)=0;
            end
        end
    end
end
```





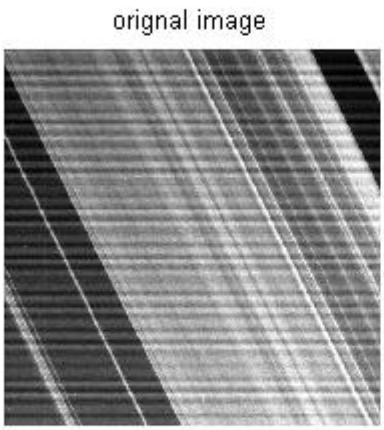


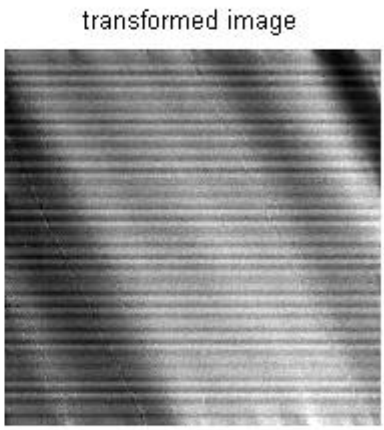
Question B6

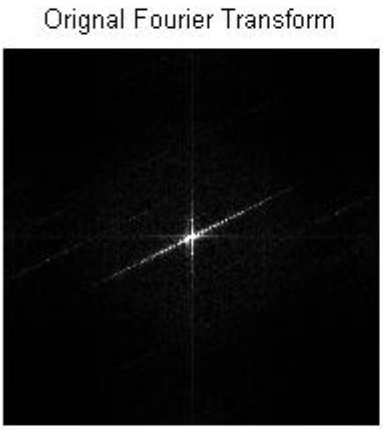
```
function []=Bquestion6()
    image=imread('PeriodicInterference.tif');
    if(size(image,3)==3)
       image=rgb2gray(image);
    end
    %figure;imshow(image);
    %imout=band_reject(image,30,330);
    imout2=band_reject2(image);
end

function[A2]=show_F(F,str)
         A=abs(F);
         max1=max(A(:));
         A2=A/max1*255;
         %figure;imshow(A2);title(str);
end

function[imout]=band_reject(image,D1,D2)

       %code goes here
       F=fft2(image);
       F=fftshift(F);
       figure;imagesc(uint8(log(1+abs(F))));colorbar;
         [s1,s2]=size(image);
         A=abs(F);
         max1=max(A(:));
         display(max1);
```

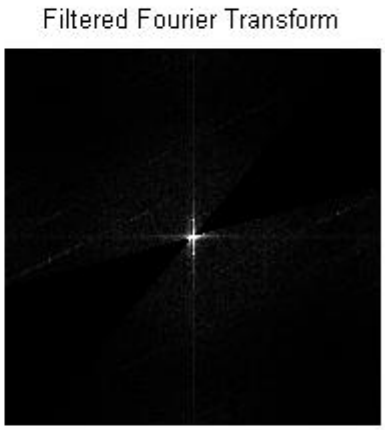
```
          A2=A/max1*255;
          display(max(A2(:)));
          figure;imshow(A2);


          for i=1:s1
              for j=1:s2
                 kip=sqrt((i-s1/2)^2+(j-s2/2)^2);
                  if(kip>=D1&&kip<=D2)
                    F(i,j)=0 ;
                 end
              end
          end
          A=abs(F);
          max1=max(A(:));
          display(max1);
          A2=A/max1*255;
          display(max(A2(:)));
          figure;imshow(A2);
        F2=ifftshift(F);
        imout=uint8(ifft2(F2));

         figure;
         subplot(1,2,1);imshow(image);title('orignal image');
         subplot(1,2,2);imshow(imout);title('Transformed image');

end

function[F]=band_reject2(image)
    [s1,s2]=size(image);
    F=fft2(image);F=fftshift(F);
    A_in=show_F(F,'orignal');
    mag=abs(F);
    mag_sorted_dec=sort(mag,'descend');
    max_value=mag_sorted_dec(1);
    threshold=0.6*max_value;
    theta1=15;theta2=50;
    for i=1:s1
        for j=1:s2
            D=(i-floor(s1/2))^2+(j-floor(s2/2))^2;
            theta=atan((i-s1/2)/(-j+s2/2))*180/pi;
            if(theta>theta1&&theta<theta2&&D>30)
                F(i,j)=0;
            end
        end
    end
    A_out=show_F(F,'transformed');
    y=uint8(abs(ifft2(F)));
    figure;
    subplot(1,2,1);imshow(image);title('orignal image');
    subplot(1,2,2);imshow(y);title('transformed image');

    figure;
    subplot(1,2,1);imshow(A_in);title('Orignal Fourier Transform');
    subplot(1,2,2);imshow(A_out);title('Filtered Fourier Transform');
```

```
end
```


orignal image

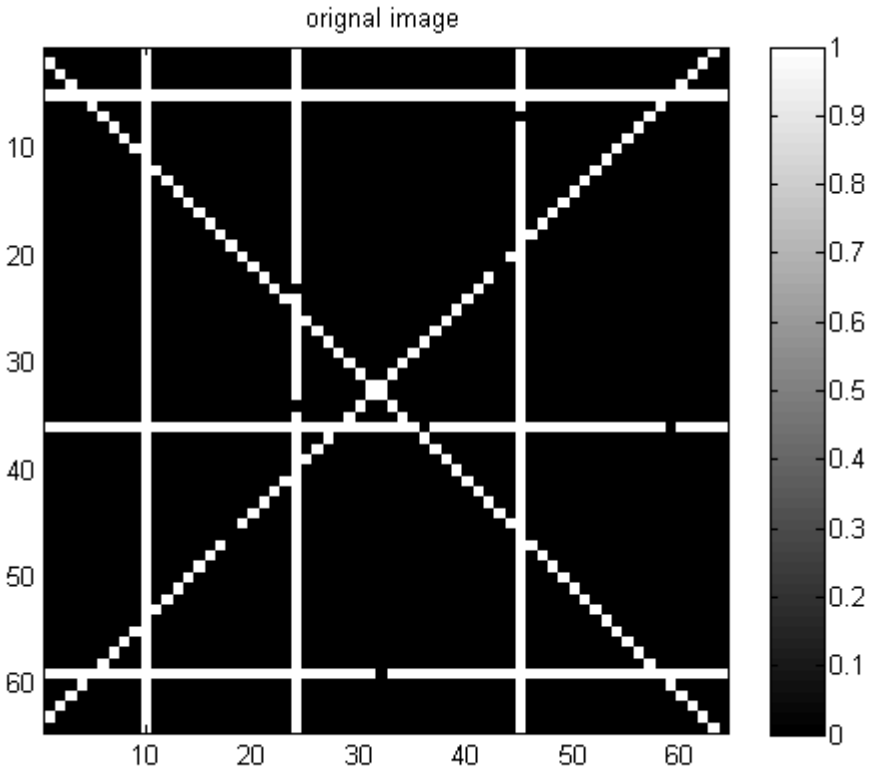

transformed image

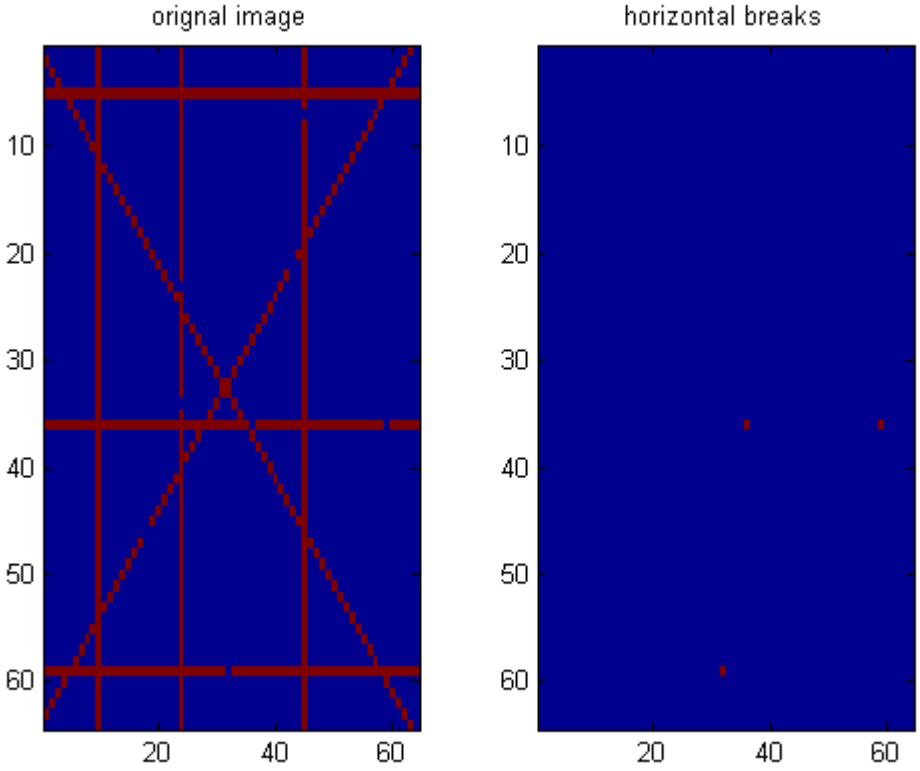

Orignal Fourier Transform

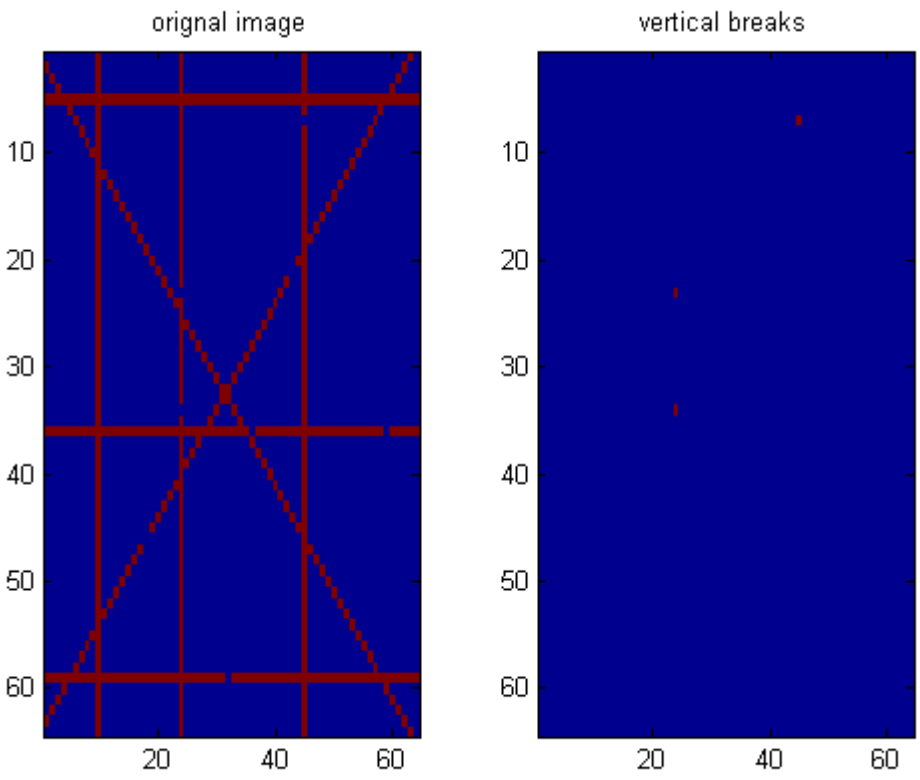

Filtered Fourier Transform

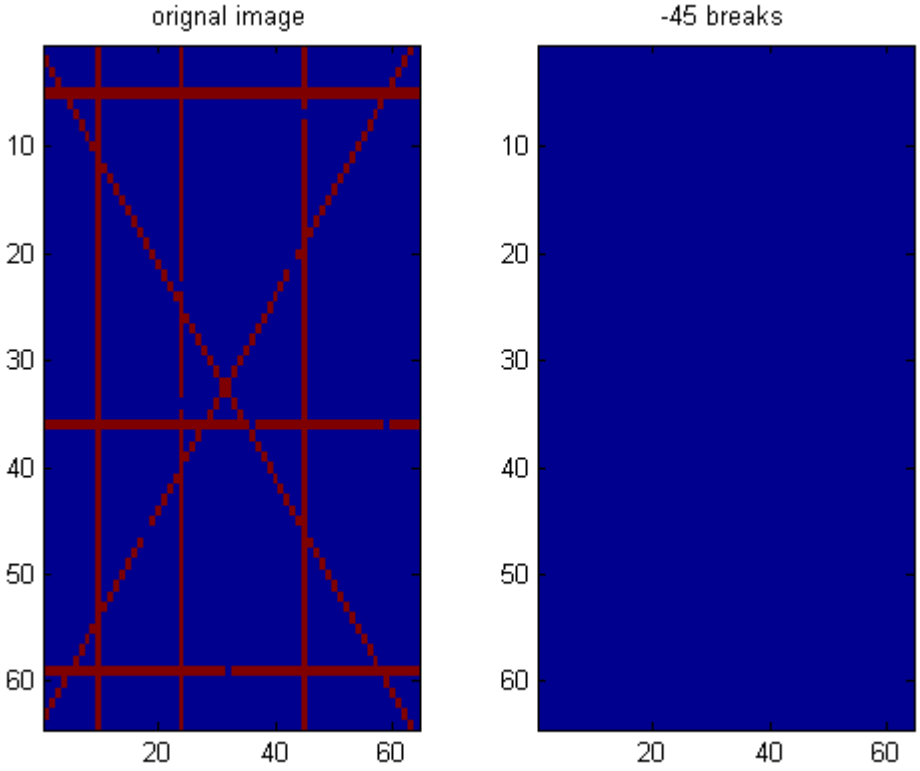
Question B7

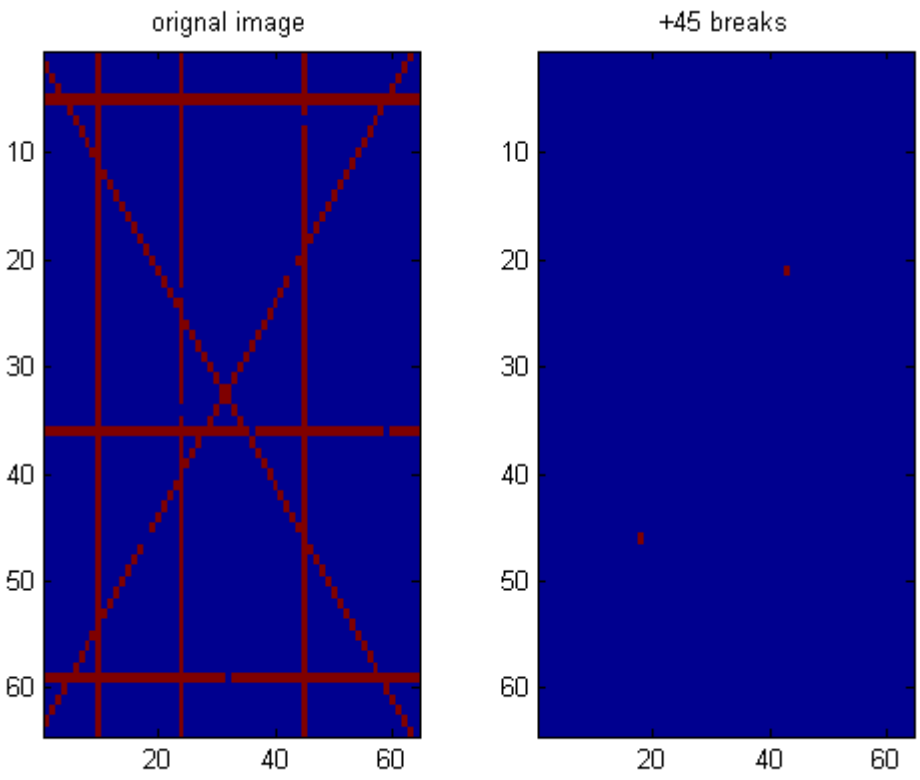
```
close all;clc
    M = 64;
f = zeros(64);
f(5,:) = 1;
f(36,:) = 1;
f(59,:) = 1;
f(:,10) = 1;
f(:,24) = 1;
f(:,45) = 1;
k = (1:M:(M^2)) + (1:M);k = k(find((k>0).*(k<=M^2)));
f(k) = ones(size(k));
k = (1:M:(M^2)) - (1:M);k = k(find((k>0).*(k<=M^2)));
f(k) = ones(size(k));
noise = imnoise(f,'salt & pepper',0.05);
f(find(noise==0)) = 0;
figure;imagesc(f);axis image;title('orignal image'); colormap gray;colorbar

for m=1:4
    [s1,s2]=size(f);
    if(m==1)
        h1=[0 0 0 0 0;0 0 0 0 0;0.25 0.25 -1 0.25 0.25;0 0 0 0 0;0 0 0 0 0];%horizontal break
    elseif(m==2)
        h1=[0 0 0.25 0 0;0 0 0.25 0 0;0 0 -1 0 0;0 0 0.25 0 0;0 0 0.25 0 0];
    elseif(m==3)
        h1=[0.25 0 0 0 0;0 0.25 0 0 0;0 0 -1 0 0; 0 0 0 0.25 0;0 0 0 0 0.25];
    elseif(m==4)
        h1=[0 0 0 0 0.25;0 0 0 0.25 0;0 0 -1 0 0; 0 0.25 0 0 0;0.25 0 0 0 0];

    end
    im1=conv2(f,h1,'same');
    for i=1:s1
        for j=1:s2
            if(im1(i,j)<1)
                im1(i,j)=0;
            end
        end
    end
    figure;
    subplot(1,2,1);imagesc(f);title('orignal image');
    subplot(1,2,2);imagesc(im1,[0,1]);%colorbar;
    if(m==1)
       title('horizontal breaks');
    elseif(m==2)
        title('vertical breaks');
    elseif(m==3)
        title('-45 breaks');
    elseif(m==4)
        title('+45 breaks');
    end
end
```


orignal image
10
20
30
40
50
60
10
20
30
40
50
60
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0


orignal image
horizontal breaks
10
20
30
40
50
60
20
40
60
10
20
30
40
50
60
20
40
60


orignal image
vertical breaks
10
20
30
40
50
60
20
40
60


orignal image
-45 breaks
10
20
30
40
50
60
20
40
60


orignal image
+45 breaks
10
20
30
40
50
60
20
40
60

Part C

Question C1

```
function[]=Cquestion1()
    h=[0 0.25 0;0.25 0 0.25;0 0.25 0];
    H=fft2(h);H_abs=abs(H);
    display(h);
    display(H_abs);
end
```

```
h =

         0    0.2500         0
    0.2500         0    0.2500
         0    0.2500         0


H_abs =

    1.0000    0.2500    0.2500
    0.2500    0.5000    0.5000
    0.2500    0.5000    0.5000
```

Question C2

Assuming an image with non zero pixel values along its borders. Padding essentially adds zero intensity values along the borders, the positions where the borders end are effectively horizontal and vertical edges in the image. This addition of edges results in higher fourier series coefficients along the x and the y axis and in shifted fft along the central horizontal and central vertical axis.

Question C3.

Assuming an image with non zero pixel values along its borders, addition of zero intensity values in padding adds horizontal and vertical edges.

Question C3:

- Consider an image $f(x,y)$ of size $M \times N$ with Fourier Spectrum = $F(u,v)$
- Step 1 → multiplying $f(x,y)$ with $(-1)^{x+y}$

$$F \xrightarrow{\text{Step1}} F_1.$$

$$F = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y)\, e^{-j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)}$$

$$F_1 = \sum\sum f(x,y) \times \underbrace{e^{-j2\pi\left(\frac{x}{2} + \frac{y}{2}\right)}}_{= (-1)^{x+y}} \times e^{-j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)}$$

$$= \sum\sum f(x,y)\, e^{-j2\pi\left(\left(\frac{u-M/2}{M}\right)x + \left(\frac{v-N/2}{N}\right)y\right)}$$

$$\boxed{F_1 = F(u-M/2, v-N/2)} \quad - \quad (A)$$

- Step2 → Computing DFT → Computing F_1
- Step3 → taking Complex conjugate

$$F_1 \xrightarrow{\text{Step3}} F_2$$

$$F_2 = F_1^*$$

$$= \sum\sum f(x,y) \left[\exp\left\{ -j2\pi \left(\left(\frac{u-M/2}{M}\right)x + \left(\frac{v-N/2}{N}\right)y \right) \right\} \right]^*$$

$$= \sum\sum f(x,y) \left[\exp\left\{ -j2\pi \left(\frac{(M/2-u)}{M}x + \left(\frac{N/2-u}{N}\right)v \right\} \right]$$

$$\boxed{F_2 = F(M/2-u, N/2-v)} \quad - (13)$$

Step 4 → Computing DFT → finding F_2

Step 5 → multiplying with $(-1)^{x+y}$

$F_2 \xrightarrow{\text{Step 5}} F_3$

Similar to step 1 where $F(u,v) \longrightarrow F(u-M/2, v-N/2)$

$$F_3(u,v) = F_2(u-M/2, v-N/2) \quad \text{Using 13}$$

$$= F(M/2-u-M/2, N/2-v-N/2)$$

$$\boxed{F_3(u,v) = F(-u,-v)} \quad - (c)$$

P.T.O.

$\therefore$ F is periodic. $F(-u,-v) = F(M-u, N-v)$

$$\therefore \boxed{F_3 = F(M-u, N-v)}$$

$$f_3 = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1}$$

$F(-u,-v) = F(M-u, N-v)$

$F(M-u, N-v) = \sum\sum f(x,y)\, e^{-j2\pi}\left(\right.$

$F_3(u,v) = F(-u,-v) = \sum\sum f(x,y)\, \exp\left\{-j2\pi\left(\frac{-ux}{m} \frac{-vy}{N}\right)\right\}$

$= \sum\sum f(-x,-y)\, \exp\left\{-j2\pi\left(\frac{ux}{m} + \frac{vy}{N}\right)\right\}$

$\because f(x,y)$ is considered periodic.
$f(-x,-y) = f(M-x, N-y)$.

$\therefore F_3(u,v) \underset{\text{Fourier Pair}}{\Longleftrightarrow} f(M-x, N-y)$

$\therefore$ the resultant image is flipped in both x and y directions

Question C4

a) Image not available.

b) The result of lowpass operation followed by high pass would be same if the order is reversed because filtering is a linear operation

Question C4

a). Low pass filtering with Gaussian function introduc a gradient in the resulting image, with brightest points at the center and lower intensities towards the boundary
When high pass filtering is done. this gradient is highlighted in the image.

Question C5

-> Question not clear