```
% Problem 3: Frequency Domain Low-pass, Band-pass and High-pass Filtering.
clear all;

% functionality of a particular functions.
% fft2() - It will return the two-dimensional fourier transform of a
% matrix using the fast fourier trtansform.
% fftshift() - It will rearrange a Fourier transform x by shifting the
% zero-frequency component to the center of the array.
% ifft2() - It will return the two-dimensional inverse fourier transform of a
% matrix using the fast fourier trtansform.

input_image = double(imread("bridge.tif")); % reading the original image. This image is↙
8 bits/pixel gray-scale
% image.
[n_rows, n_cols] = size(input_image); % It will return a x * y dimension of the image.↙
x represents the number of
% rows in the image and y represents the number of columns of a matrix formed from the↙
input image.
% for example if we have a 3-by-4 image then size function will return [3 4].

% The 2-D DFT of the original image and shifting the DC component.
input_image_2ddft = fft2(input_image); % 2D-DFT of the original image.
input_image_2ddft_shift = fftshift(input_image_2ddft); % Shifting of the DC component.

%center of image
mid_value = n_rows/2; % finding the middle of the image.

% We want to do the indexing of the matrix, so we will create rectuangluar
% structures from the given array. It is basically done with the help of
% meshgrid function.
[vector_col,vector_row] = meshgrid(1:n_cols,1:n_rows); % It will return 2-D grid↙
coordinates based on the
% coordinates in vectors (1:n_row) and (1:n_col).

% Evaluating distance over the grid. Calculating the euclidean distance(distance↙
between two points).
distance = ((vector_row-mid_value).^2+(vector_col-mid_value).^2).^0.5; % simply↙
calculating the distance between
% two points.

% Here, we will design 3 filters in total. 1] Low-pass filter, 2] high-pass
% filter, and 3] bandpass filter.

% Designing low pass filter

distance_low =  (1/8) * mid_value; % It is given in the problem statement that↙
frequency response of the ideal
% low pass filter is equal to one-eighth of the distance from the center to the↙
horizontal or vertical edge.
low_pass_filter = (distance <= distance_low); % here the distance must be less than or↙
```

```
equal to the frequency
% response of the ideal low pass filter.
low_pass_filtered_image = input_image_2ddft_shift.*low_pass_filter; % Here, the '.'↙
after input_image_2ddft_shift
% indicates that it will take all the pixel values. i.e. it will take all the values of↙
each and every rows and
% columns from the matrix. Also, we are applying a low pass filter on the shifted↙
image.
low_pass_filtered_spatial_image = real(ifft2(fftshift(low_pass_filtered_image))); %↙
Here we are applying inverse
% 2D-DFT on the low pass filtered image to obtain the image in spatial domain. % So,↙
first we apply fftshift and
% then ifft2 function for inverse 2D-DFT.

% Designing high pass filter

distance_high = (1/2) * mid_value; % It is given in the problem statement that↙
frequency response of the ideal
% high pass filter is equal to one-twoth of the distance from the center to the↙
horizontal or vertical edge.
high_pass_filter = (distance >= distance_high); % here the distance must be greater↙
than or equal to the frequency
% response of the ideal high pass filter.
high_pass_filtered_image = input_image_2ddft_shift.*high_pass_filter; % Here, the '.'↙
after input_image_2ddft_shift
% indicates that it will take all the pixel values. i.e. it will take all the values of↙
each and every rows and
% columns from the matrix. Also, we are applying a high pass filter on the shifted↙
image.
high_pass_filtered_spatial_image = real(ifft2(fftshift(high_pass_filtered_image))); %↙
Here we are applying inverse
% 2D-DFT on the high pass filtered image to obtain the image in spatial domain. % So,↙
first we apply fftshift and
% then ifft2 function for inverse 2D-DFT.

% Deigning band pass filter

band_pass_filter = (distance >= distance_low & distance <= distance_high); % here the↙
distance must be greater
% than or equal to the frequency response of the ideal low pass filter and less than or↙
equal to the frequency
% response of the ideal high pass filter.
band_pass_filtered_image = input_image_2ddft_shift.*band_pass_filter; % Here, the '.'↙
after input_image_2ddft_shift
% indicates that it will take all the pixel values. i.e. it will take all the values of↙
each and every rows and
% columns from the matrix. Also, we are applying a band pass filter on the shifted↙
image.
band_pass_filtered_spatial_image = real(ifft2(fftshift(band_pass_filtered_image)));  %↙
Here we are applying inverse
```

```
% 2D-DFT on the band pass filtered image to obtain the image in spatial domain. % So,↙
first we apply fftshift and
% then ifft2 function for inverse 2D-DFT.

f = figure; % figure creates figure graphics objects. figure objects are the individual↙
windows on the screen
% in which MATLAB displays graphical output.
subplot(2,2,1); % subplot(m, n, p) divides the current figure into an m-by-n grid and↙
creates axes in the
% position specified by p. Here, m=n=2, p=1.
imshow(uint8(input_image)); % It will display the gray-scale image in the figure and
% it will convert each and every pixel value of the input image into the range of 0 to↙
255.
title('Original_Image'); % It will add the specified title for the current plot.
subplot(2,2,2); % subplot(m, n, p) divides the current figure into an m-by-n grid and↙
creates axes in the
% position specified by p. Here, m=n=p=2.
imshow(low_pass_filtered_spatial_image,[]); % It will display the gray-scale image in↙
the figure.
title('Low-pass filtered Image'); % It will add the specified title for the current↙
plot.
subplot(2,2,3); % subplot(m, n, p) divides the current figure into an m-by-n grid and↙
creates axes in the
% position specified by p. Here, m=n=2, p=3.
imshow(band_pass_filtered_spatial_image,[]); % It will display the gray-scale image in↙
the figure.
title('Band-pass filtered Image'); % It will add the specified title for the current↙
plot.
subplot(2,2,4); % subplot(m, n, p) divides the current figure into an m-by-n grid and↙
creates axes in the
% position specified by p. Here, m=n=2, p=4.
imshow(high_pass_filtered_spatial_image,[]); % It will display the gray-scale image in↙
the figure.
title('High-pass filtered Image'); % It will add the specified title for the current↙
plot.

f = figure; % figure creates figure graphics objects. figure objects are the individual↙
windows on the screen
% in which MATLAB displays graphical output.
max_amplitude = max(input_image(:)); % It will simply find the maximum amplitude of the↙
image.
log_shifted_img = log(1+abs(input_image_2ddft_shift)); % Apply log transform on the↙
2ddft shifted image
% and make it log shifted image.
subplot(2,2,1); % subplot(m, n, p) divides the current figure into an m-by-n grid and↙
creates axes in the
% position specified by p. Here, m=n=2, p=1.
imshow(log_shifted_img./max_amplitude,[]); % It will display the gray-scale image in↙
the figure.
title('Original Image'); % It will add the specified title for the current plot.
```

```
log_shifted_low = log(1+abs(low_pass_filtered_image)); % Apply log transform on the↙
2ddft shifted image
% and make it log shifted image.
subplot(2,2,2); % subplot(m, n, p) divides the current figure into an m-by-n grid and↙
creates axes in the
% position specified by p. Here, m=n=p=2.
imshow(log_shifted_low./max_amplitude, []); % It will display the gray-scale image in↙
the figure.
title('Low-pass filtered'); % It will add the specified title for the current plot.
log_shifted_band = log(1+abs(band_pass_filtered_image)); % Apply log transform on the↙
2ddft shifted image
% and make it log shifted image.
subplot(2,2,3); % subplot(m, n, p) divides the current figure into an m-by-n grid and↙
creates axes in the
% position specified by p. Here, m=n=2, p=3.
imshow(log_shifted_band./max_amplitude, []); % It will display the gray-scale image in↙
the figure.
title('Band-pass filtered'); % It will add the specified title for the current plot.
log_shifted_high = log(1+abs(high_pass_filtered_image)); % Apply log transform on the↙
2ddft shifted image
% and make it log shifted image.
subplot(2,2,4); % subplot(m, n, p) divides the current figure into an m-by-n grid and↙
creates axes in the
% position specified by p. Here, m=n=2, p=4.
imshow(log_shifted_high./max_amplitude, []); % It will display the gray-scale image in↙
the figure.
title('High-pass filtered'); % It will add the specified title for the current plot.
```