Consider this old photo affected by periodic noise. You have to develop a method able to extract some statistics about related to the noise by analyzing small patches of the image and then use this information in order to perform denoising in the frequency domain.

Write a MATLAB script able to perform the following operations:

a) Read the grayscale image *'input_image.png'*, convert it into a double representation and visualize it.
b) Consider 16 non overlapping patches with equal size, obtained with a 4x4 grid, and for each one of them preform the following operations:
    I. Obtain the frequency domain representation of the considered patch using FFT and shift it in order to have the DC component in the center
    II. Compute the power spectrum and store the result in **PS_patch_array(:,:,k)** (where k is index of the considered patch)
c) Apply the FFT on the original input image, shift the result in order to have the DC component in the center, obtaining **DFT_original**
d) Compute the power spectrum of the input image and save it as **PS**
e) Compute the median value on the third axis of **PS_patch_array** in order to obtain **PS_patch**
f) Resize **PS_patch** in order to match the size of **PS**
g) Apply a median filter on **PS_patch** using a 25x25 filter, obtaining **PS_patch_med**
h) Obtain a binary image, called **Mask**, with the same size of **PS** having true values in each frequency coordinate in which **PS_patch** > 3 * **PS_patch_med**
i) Build **DFT_restored** as follow:
    I. For low frequency coordinates just copy the corresponding value of **DFT_original** (use 80 as the threshold distance from DC that defines low frequency)
    II. For high frequencies assign 0 if **Mask** is true in that location or the corresponding value of **DFT_original** otherwise
j) Obtain the restore image applying the inverse FFT to **DFT_restored** and show the result.

```
clc
close all
clear all

%a)
img = im2double(imread('input_image.png'));
figure; imshow(img);

%b)
n = 4;
h = size(img,1)/n;
w = size(img,2)/n;
k = 0;

for i=1:h:size(img,1)
    for j=1:w:size(img,2)
        k = k+1;
        img_patch = img( i:(i+h-1), j:(j+w-1) );
        %b).I
        DFT_patch = fft2(img_patch);
        DFT_patch = fftshift(DFT_patch);
        %b).II
        PS = real(DFT_patch).^2 + imag(DFT_patch).^2;
        PS_patch_array(:,:,k) = PS;
    end
end

%c)
DFT_original = fft2(img);
DFT_original = fftshift(DFT_original);

%d)
PS = real(DFT_original).^2 + imag(DFT_original).^2;

%e)
PS_patch = median(PS_patch_array,3);

%f)
PS_patch = imresize(PS_patch,size(PS));

%g)
PS_patch_med = medfilt2(PS_patch,[25 25]);

%h)
Mask = PS_patch>3*PS_patch_med;

%i)
DFT_restored = DFT_original;
mid = [1 + size(DFT_original,1)/2,1 + size(DFT_original,2)/2];
D0 = 80;
for i=1:size(DFT_original,1)
    for j=1:size(DFT_original,2)
        r = sqrt((i-mid(1))^2+(j-mid(2))^2);
        %i).I
        if r<D0
              DFT_restored(i,j) = DFT_original(i,j);
        %i).II
        else
              if(Mask(i,j) == 1)
                  DFT_restored(i,j) = 0;
              end
        end
    end
end

%j)
img_restored = (ifft2(ifftshift(DFT_restored), 'symmetric'));
figure; imshow(img_restored);
```