```
clc;
clear all;
close all;

I = imread("input.jpg");
if size(I,3) == 3
    I = rgb2gray(I);
end
I = double(I);

Q = [16 11 10 16 24 40 51 61;
     12 12 14 19 26 58 60 55;
     14 13 16 24 40 57 69 56;
     14 17 22 29 51 87 80 62;
     18 22 37 56 68 109 103 77;
     24 35 55 64 81 104 113 92;
     49 64 78 87 103 121 120 101;
     72 92 95 98 112 100 103 99];
% JPEG standard quantization table used to reduce DCT coefficients.

factor = 5;
Q = Q * factor;
% Scaling the quantization matrix to increase compression level.

blockSize = 8;
% Image is processed block-wise using 8x8 blocks as per JPEG standard.

[m,n] = size(I);
reconstructed = zeros(m,n);
% Empty matrix to store the reconstructed image blocks.

for i = 1:blockSize:(m-blockSize+1)
    for j = 1:blockSize:(n-blockSize+1)

        block = I(i:i+7, j:j+7);
        % Selecting an 8x8 block from the image.

        block = block - 128;
        % Level shifting to center pixel values around zero.

        dctBlock = dct2(block);
        % Applying DCT to convert spatial data into frequency
 components.

        quantBlock = round(dctBlock ./ Q);
        % Quantizing the DCT coefficients (main compression step).

        dequantBlock = quantBlock .* Q;
        % De-quantization during reconstruction.

        idctBlock = idct2(dequantBlock);
        % Converting frequency domain data back to spatial domain.
        idctBlock = idctBlock + 128;
        % Shifting values back to valid image intensity range.

        reconstructed(i:i+7, j:j+7) = idctBlock;
        % Storing the processed block into output image.

    end
end

reconstructed = uint8(reconstructed);
% Converting the reconstructed image to displayable format.

figure;
imshow(uint8(I));
title("Original Image");
% Showing original image before compression.

figure;
imshow(reconstructed);
title("Reconstructed Image After JPEG Compression");

% Showing image after DCT-based JPEG compression.
```

Reconstructed Image After JPEG Compression

Published with MATLAB® R2021a