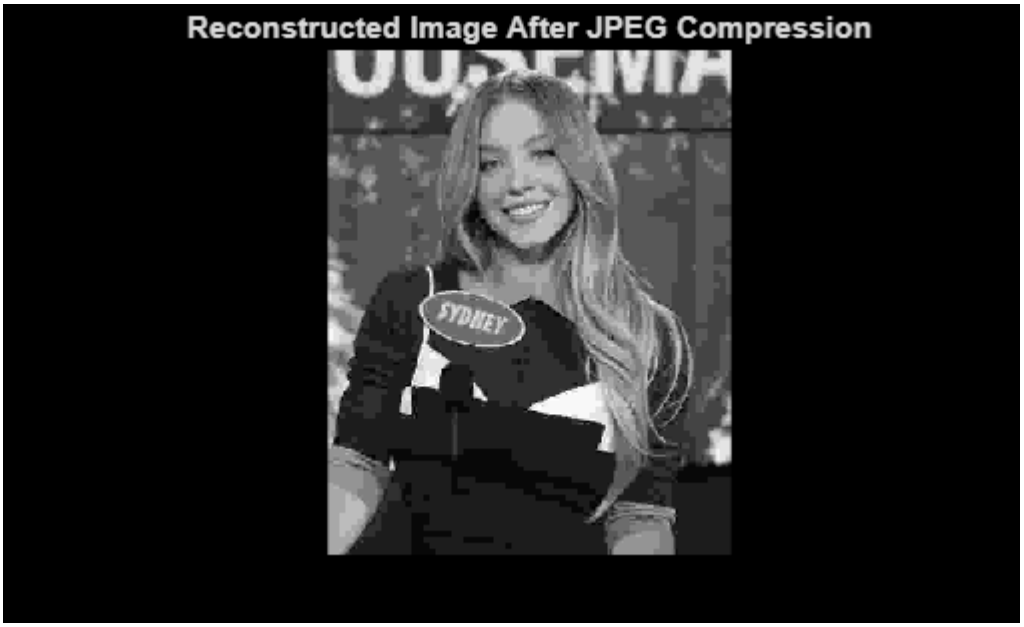
```
clc;
clear;
close all;

imgInput = imread("SS.jpeg");

% Convert to grayscale if RGB
if size(imgInput,3) == 3
    imgInput = rgb2gray(imgInput);
end

imgInput = double(imgInput);
% Load grayscale image and convert to double for DCT processing

quantMatrix = [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];
% Standard JPEG quantization table

scaleValue = 10;
quantMatrix = quantMatrix * scaleValue;
% Increase quantization strength for higher compression

blk = 8;
% JPEG uses fixed 8×8 blocks

[row, col] = size(imgInput);
outputImg = zeros(row, col);
% Empty matrix for reconstructed image

for r = 1:blk:(row - blk + 1)
    for c = 1:blk:(col - blk + 1)

        imgBlock = imgInput(r:r+7, c:c+7);
        % Extract 8×8 block

        imgBlock = imgBlock - 128;
        % Level shift around zero

        freqBlock = dct2(imgBlock);
        % Apply DCT

        quantized = round(freqBlock ./ quantMatrix);
        % Quantization (lossy step)

        dequantized = quantized .* quantMatrix;
        % De-quantization
```

```
        spatialBlock = idct2(dequantized);
        % Inverse DCT

        spatialBlock = spatialBlock + 128;
        % Shift back intensity range

        outputImg(r:r+7, c:c+7) = spatialBlock;
        % Store reconstructed block

    end
end

outputImg = uint8(outputImg);
% Convert back to uint8 for display

figure;
imshow(uint8(imgInput));
title("Original Image");

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



Published with MATLAB® R2025b