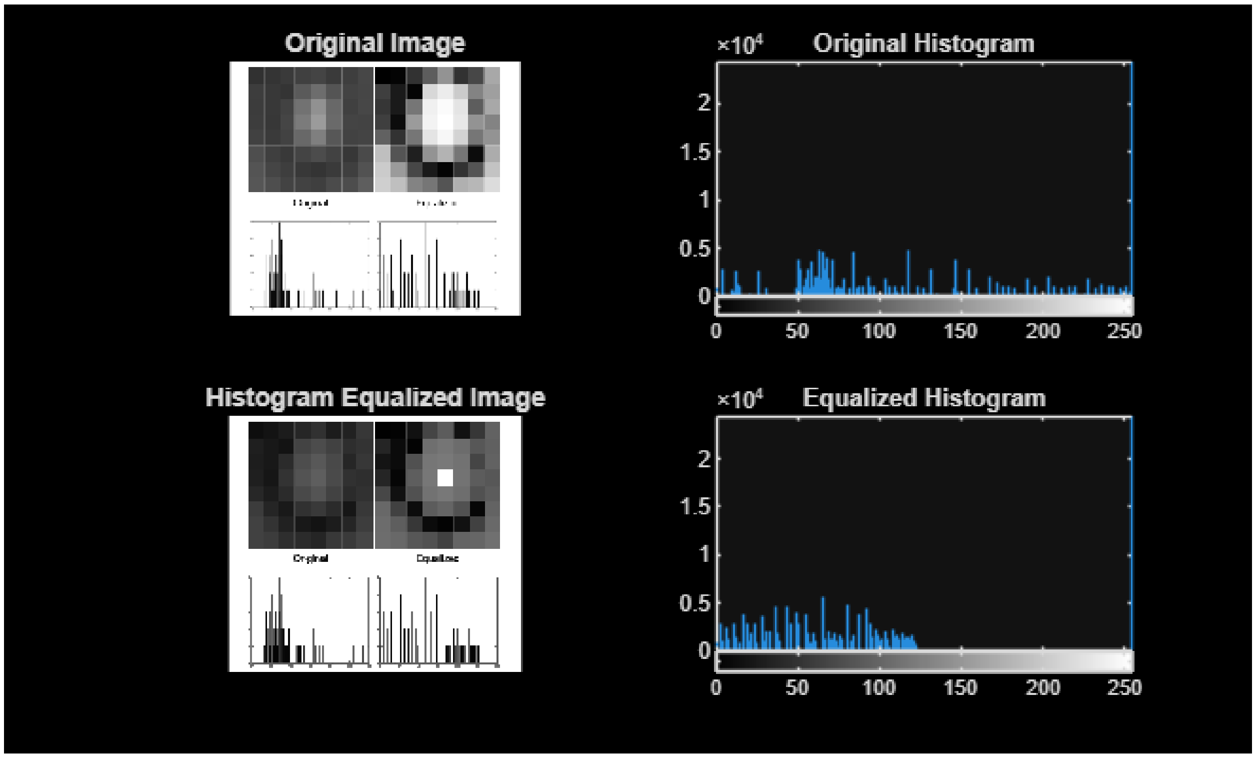
```
clc;
clear all;
close all;
I=imread("Hist.png");
if size(I,3) == 3
    I = rgb2gray(I);
end
%to convert the image to grayscale if it is RGB
%%The below code provides the direct keywords to find the desired output
% J=histeq(I);
% figure;
% subplot(2,2,1);
% imshow(I);
% title('Original Grayscale Image');
%
% subplot(2,2,2);
% imhist(I);
% title('Original Histogram');
%
% subplot(2,2,3);
% imshow(J);
% title('Histogram Equalized Image');
%
% subplot(2,2,4);
% imhist(J);
% title('Equalized Histogram');
%%the below code gives the mathematical approach for the desired outcome
[M,N]=size(I); %Calculating the size of the image
numPixels=M*N; %multiplying the rows and columns for total number of pixels
hist=zeros(256,1); %creating a row matrix for the pixel intensity for 0 to 256
for i=1:M
for j=1:N
        intensity=I(i,j);
        hist(intensity+1)=hist(intensity+1)+1;
end
end
%Loop to calculate each and every pixel and add the number of pixels at a
%certain intensity in the row matrix with that value.
pdf=hist/numPixels; %normalising or calculating the average of the intensity
of pixels
cdf = zeros(256,1);
cdf(1) = pdf(1);
for k = 2:256
    cdf(k) = cdf(k-1) + pdf(k);
end
%calclating the cumulative frequency distribution of the given image
1
L = 256;
mapping = round((L - 1) * cdf);
%mapping the new intensity values with respect to the old frequency values
I_eq = zeros(M,N,'uint8'); %typecasting to the correct format
for i = 1:M
for j = 1:N
        oldVal = I(i,j);
        I_eq(i,j) = mapping(oldVal + 1);
end
end
%putting the new equalised values in the image format
figure;
subplot(2,2,1);
imshow(I);
title('Original Image');
subplot(2,2,2);
imhist(I);
title('Original Histogram');
subplot(2,2,3);
imshow(I_eq);
title('Histogram Equalized Image');
subplot(2,2,4);
imhist(I_eq);
title('Equalized Histogram');
```

```
ans =

     1
```



*Published with MATLAB® R2025b*