```matlab
clc;
clear;
close all;

% Read grayscale image
img = imread('img.jpeg');
if size(img,3) == 3
    img = rgb2gray(img);
end
img = uint8(img);

% Flatten image into 1D array
pixels = img(:);

% Compute symbol frequencies
symbols = unique(pixels);
freq = zeros(length(symbols),1);

for i = 1:length(symbols)
    freq(i) = sum(pixels == symbols(i));
end

% Sort symbols by descending frequency
[~, idx] = sort(freq, 'descend');
symbols = symbols(idx);
freq = freq(idx);

% Initialize Shannon-Fano code map
codes = containers.Map('KeyType','double','ValueType','char');
for i = 1:length(symbols)
    codes(symbols(i)) = '';
end

% Generate Shannon-Fano codes
codes = shannonFanoRecursive(symbols, freq, codes);

% Encode image
encoded_bits = '';
for i = 1:length(pixels)
    encoded_bits = [encoded_bits codes(double(pixels(i)))];
end

% Decode bitstream
decoded_pixels = [];
current_bits = '';

% Reverse code map
code_keys = keys(codes);
code_values = values(codes);
reverse_codes = containers.Map(code_values, code_keys);

for i = 1:length(encoded_bits)
```

```matlab
        current_bits = [current_bits encoded_bits(i)];
        if isKey(reverse_codes, current_bits)
            decoded_pixels(end+1) = reverse_codes(current_bits); %#ok<SAGROW>
            current_bits = '';
        end
    end
end

decoded_img = reshape(uint8(decoded_pixels), size(img));

% Display results
figure;
subplot(1,2,1);
imshow(img);
title('Original Image');
axis off;

subplot(1,2,2);
imshow(decoded_img);
title('Decoded Image (Shannon-Fano)');
axis off;

% Compression statistics
original_size_bits   = numel(pixels) * 8;
compressed_size_bits = length(encoded_bits);
compression_ratio    = original_size_bits / compressed_size_bits;

fprintf('\n--- Compression Statistics ---\n');
fprintf('Original size (bits):   %d\n', original_size_bits);
fprintf('Compressed size (bits): %d\n', compressed_size_bits);
fprintf('Compression Ratio:      %.2f\n', compression_ratio);
fprintf('-----------------------------\n');

% Local function: Shannon-Fano recursive split
function codes = shannonFanoRecursive(symbols, freq, codes)

    % Base case
    if length(symbols) <= 1
        return;
    end

    % Total frequency
    total_freq = sum(freq);

    % Find split point
    acc_freq = 0;
    split_index = 1;

    for i = 1:length(freq)
        acc_freq = acc_freq + freq(i);
        if acc_freq >= total_freq / 2
            split_index = i;
            break;
        end
    end
```

```matlab
    % Assign 0 to first group
    for i = 1:split_index
        codes(symbols(i)) = [codes(symbols(i)) '0'];
    end

    % Assign 1 to second group
    for i = split_index+1:length(symbols)
        codes(symbols(i)) = [codes(symbols(i)) '1'];
    end

    % Recursive calls
    codes = shannonFanoRecursive(symbols(1:split_index), ...
                                 freq(1:split_index), codes);

    codes = shannonFanoRecursive(symbols(split_index+1:end), ...
                                 freq(split_index+1:end), codes);
end


--- Compression Statistics ---
Original size (bits):   237600
Compressed size (bits): 77692
Compression Ratio:      3.06
------------------------------
```


Original Image   Decoded Image (Shannon–Fano)

Published with MATLAB® R2025b