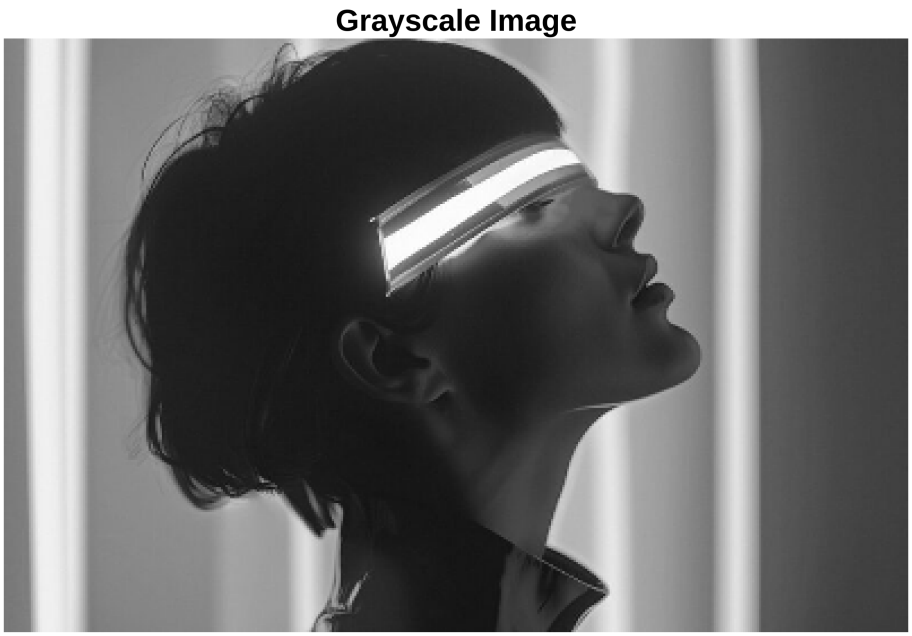
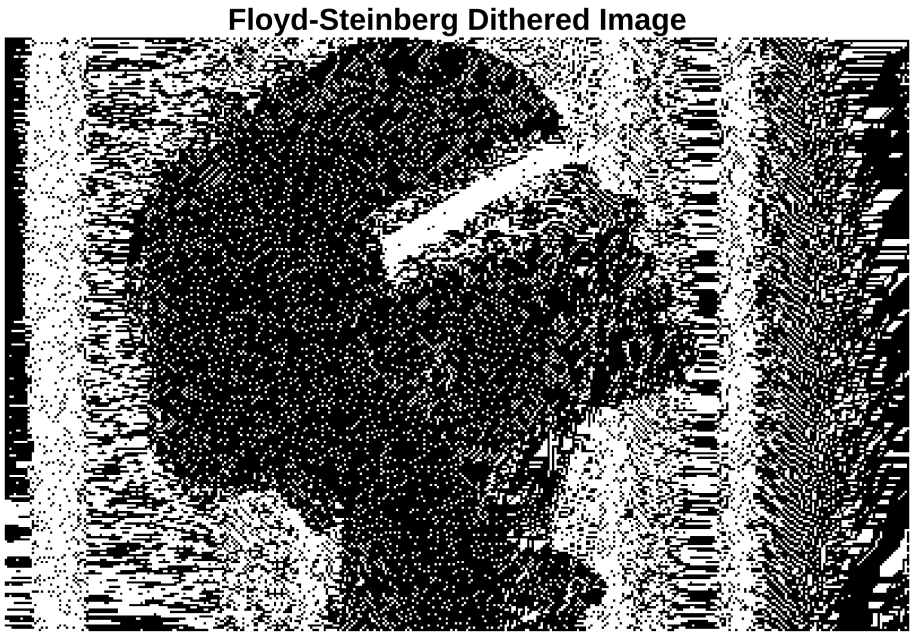
```
% Load the image
img = imread('DIP_assign_img.jpg'); % Use the uploaded image filename

% Convert to grayscale if the image is RGB
if size(img, 3) == 3
    img_gray = rgb2gray(img);
else
    img_gray = img;
end

% Normalize the grayscale image to [0, 1]
img_gray = double(img_gray) / 255;

% Display the grayscale image
figure, imshow(img_gray), title('Grayscale Image');
```



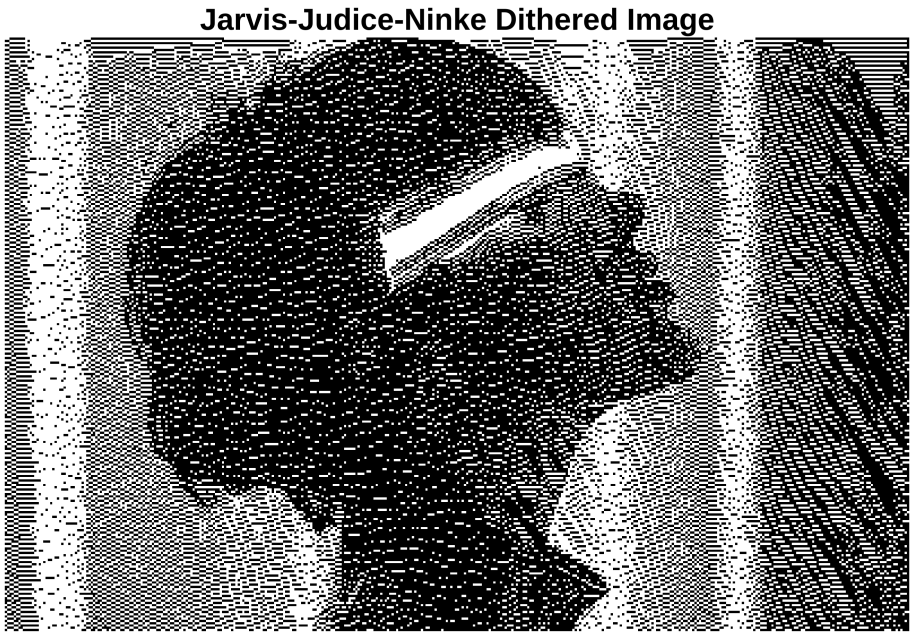

```
function img_fs = floyd_steinberg_dithering(img)
    [rows, cols] = size(img);
```

```
    img_fs = img; % Copy the original image for processing

    for y = 1:rows
        for x = 1:cols
            old_pixel = img_fs(y, x);
            new_pixel = round(old_pixel); % Quantize the pixel
            img_fs(y, x) = new_pixel;
            quant_error = old_pixel - new_pixel;

            % Diffuse the error to the neighboring pixels
            if x+1 <= cols
                img_fs(y, x+1) = img_fs(y, x+1) + quant_error * 7/16;
            end
            if x-1 > 0 && y+1 <= rows
                img_fs(y+1, x-1) = img_fs(y+1, x-1) + quant_error * 3/16;
            end
            if y+1 <= rows
                img_fs(y+1, x) = img_fs(y+1, x) + quant_error * 5/16;
            end
            if x+1 <= cols && y+1 <= rows
                img_fs(y+1, x+1) = img_fs(y+1, x+1) + quant_error * 1/16;
            end
        end
    end
end

% Apply Floyd-Steinberg dithering
img_floyd = floyd_steinberg_dithering(img_gray);

% Display the Floyd-Steinberg dithered image
figure, imshow(img_floyd), title('Floyd-Steinberg Dithered Image');
```




```
function img_jjn = jarvis_judice_ninke_dithering(img)
    [rows, cols] = size(img);
    img_jjn = img; % Copy the original image for processing

    % Jarvis-Judice-Ninke diffusion matrix
    jjn_matrix = [
        0 0 0 7 5;
        3 5 7 5 3;
        1 3 5 3 1
    ] / 48;

    % Apply the JJN error diffusion
    for y = 1:rows
        for x = 1:cols
            old_pixel = img_jjn(y, x);
            new_pixel = round(old_pixel);
            img_jjn(y, x) = new_pixel;
            quant_error = old_pixel - new_pixel;
```

```
            % Diffuse the error to the neighboring pixels
            for j = 1:3
                for i = -2:2
                    if y+j <= rows && x+i > 0 && x+i <= cols
                        img_jjn(y+j, x+i) = img_jjn(y+j, x+i) + quant_error
* jjn_matrix(j, i+3);
                    end
                end
            end
        end
    end
end

% Apply Jarvis-Judice-Ninke dithering
img_jjn = jarvis_judice_ninke_dithering(img_gray);

% Display the JJN dithered image
figure, imshow(img_jjn), title('Jarvis-Judice-Ninke Dithered Image');
```




```
% Display side-by-side comparison
figure,
subplot(1, 2, 1), imshow(img_floyd), title('Floyd-Steinberg Dithering');
subplot(1, 2, 2), imshow(img_jjn), title('Jarvis-Judice-Ninke Dithering');
```

Floyd-Steinberg Dithering

Jarvis-Judice-Ninke Dithering