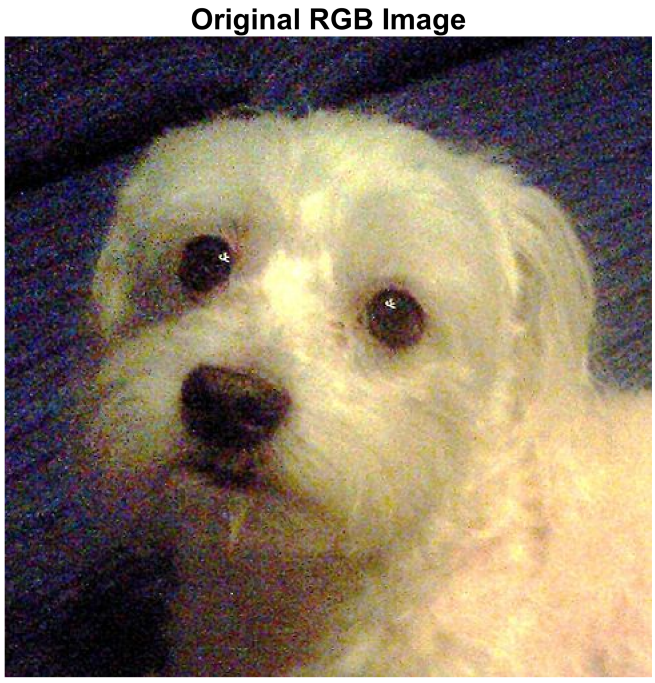
```
%Encryption Process starts
%Encryption Process-1 (Permutation)
% Read the original RGB image and convert to grayscale
rgb_image =
imread("C:\Users\shell\Desktop\485px-2._Mestre_(Italy),_the_dog_MILO.jpg");
gray_image = rgb2gray(rgb_image);
[rows, cols] = size(gray_image);
spiral_image = zeros(rows, cols, 'uint8');

% Directions: right = 1, down = 2, left = -1, up = -2
direction = 1;
r = 1;
c = 1;
rmin = 1;
rmax = rows;
cmin = 1;
cmax = cols;

% Create the spiral ordered image
for i = 1:rows * cols
    spiral_image(r, c) = gray_image(i);
    if direction == 1 % moving right
        if c < cmax
            c = c + 1;
        else
            direction = 2;
            rmin = rmin + 1;
            r = r + 1;
        end
    elseif direction == 2 % moving down
        if r < rmax
            r = r + 1;
        else
            direction = -1;
            cmax = cmax - 1;
            c = c - 1;
        end
    elseif direction == -1 % moving left
        if c > cmin
            c = c - 1;
        else
            direction = -2;
            rmax = rmax - 1;
            r = r - 1;
        end
    elseif direction == -2 % moving up
        if r > rmin
            r = r - 1;
        else
            direction = 1;
```

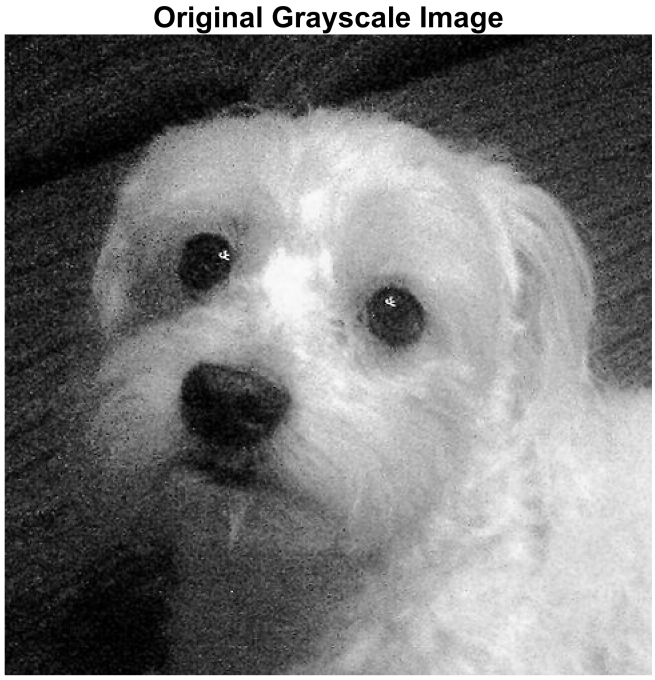
```
            cmin = cmin + 1;
            c = c + 1;
        end
    end
end
%Encryption Process-2 (Scrambling)
[rows, cols, ~] = size(spiral_image);
num_pixels = rows * cols;
secret_key = 296;
rng(secret_key);
pixel_permutation = randperm(num_pixels);
scrambled_image = spiral_image;
for i = 1:num_pixels
    original_row = ceil(pixel_permutation(i) / cols);
    original_col = mod(pixel_permutation(i) - 1, cols) + 1;
    scrambled_row = ceil(i / cols);
    scrambled_col = mod(i - 1, cols) + 1;
    scrambled_image(scrambled_row, scrambled_col, :) = spiral_image(original_row,
original_col, :);
end
%Encrytpion Process-3 (BITXOR)
s1 = size(scrambled_image, 1);
s2 = size(scrambled_image, 2);
encimg = double(scrambled_image);
secretkey = input('Enter secret key number: ');
secretcopy = secretkey;
matrix = double(ones(1, s2) * secretkey);
for i = 2:s1
    secretkey = mod(secretkey*15,256);
    rem = double(ones(1, s2) * secretkey);
    matrix = [matrix; rem];
end
encimg = uint8(encimg);
matrix = uint8(matrix);
encryptedimage = bitxor(encimg, matrix);


%Decryption Process starts

%Decryption Process-1
decryptedimage=decrypt_image(encryptedimage,secretcopy);
function decrypted_image = decrypt_image(encrypted_image, secretkey)
    s1 = size(encrypted_image, 1);
    s2 = size(encrypted_image, 2);
    encimg = double(encrypted_image);
    matrix = double(ones(1, s2) * secretkey);

    for i = 2:s1
        secretkey = mod(secretkey * 15,256);
        rem = double(ones(1, s2) * secretkey);
```

```
        matrix = [matrix; rem];
    end

    encimg = uint8(encimg);
    matrix = uint8(matrix);
    decrypted_image = bitxor(encimg, matrix);
end

%Decryption Process 2
[rows, cols, ~] = size(decryptedimage);
num_pixels = rows * cols;
rng(secret_key);
pixel_permutation = randperm(num_pixels);
constructed_image = decryptedimage;
for i = 1:num_pixels
    original_row = ceil(pixel_permutation(i) / cols);
    original_col = mod(pixel_permutation(i) - 1, cols) + 1;
    scrambled_row = ceil(i / cols);
    scrambled_col = mod(i - 1, cols) + 1;
    constructed_image(original_row, original_col, :) =
decryptedimage(scrambled_row, scrambled_col, :);
end
finalconstructed=constructed_image;
%Decryption Process-3
reconstructed_image = zeros(rows, cols, 'uint8');
direction = 1;
r = 1;
c = 1;
rmin = 1;
rmax = rows;
cmin = 1;
cmax = cols;
for i = 1:rows * cols
   reconstructed_image(i) = finalconstructed(r, c);
    if direction == 1 % moving right
        if c < cmax
            c = c + 1;
        else
            direction = 2;
            rmin = rmin + 1;
            r = r + 1;
        end
    elseif direction == 2 % moving down
        if r < rmax
            r = r + 1;
        else
            direction = -1;
            cmax = cmax - 1;
            c = c - 1;
        end
```

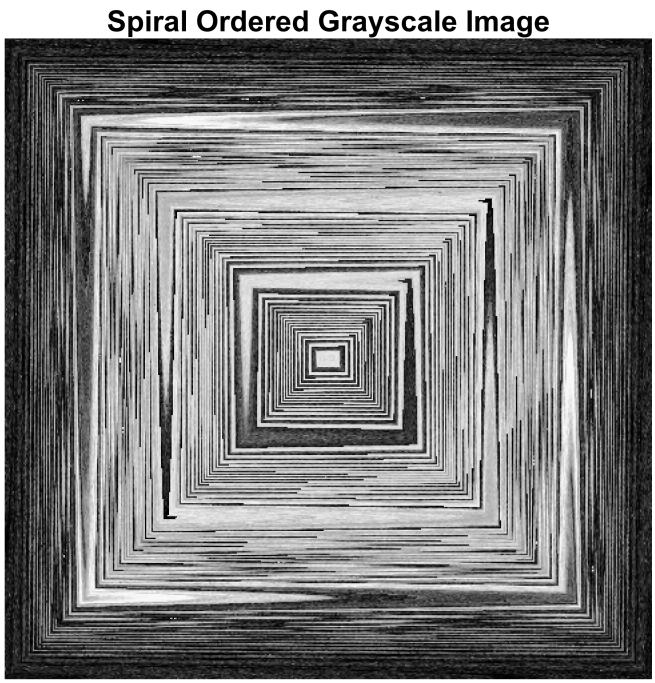
```
    elseif direction == -1 % moving left
        if c > cmin
            c = c - 1;
        else
            direction = -2;
            rmax = rmax - 1;
            r = r - 1;
        end
    elseif direction == -2 % moving up
        if r > rmin
            r = r - 1;
        else
            direction = 1;
            cmin = cmin + 1;
            c = c + 1;
        end
    end
end

figure;

imshow("C:\Users\shell\Desktop\485px-2._Mestre_(Italy),_the_dog_MILO.jpg");
title('Original RGB Image');
```



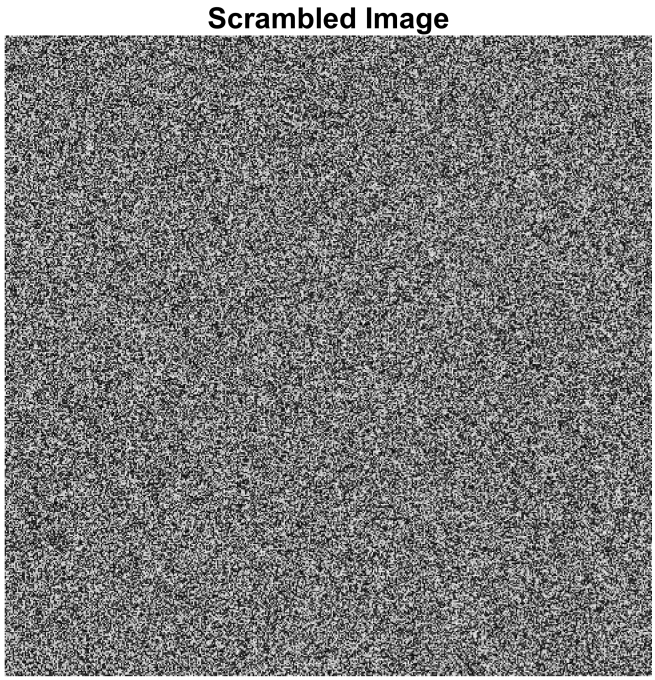

```
imshow(gray_image);
title('Original Grayscale Image');
```




```
imshow(spiral_image);
title('Spiral Ordered Grayscale Image');
```

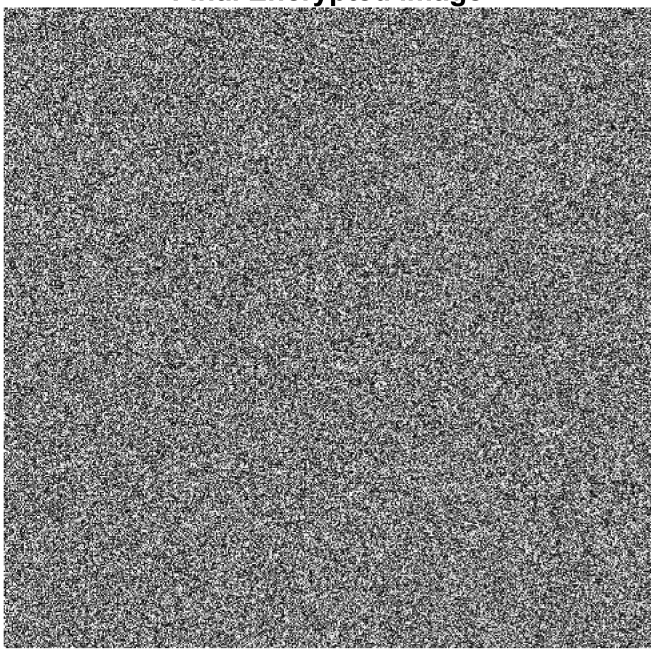


```
imshow(scrambled_image);
title('Scrambled Image');
```



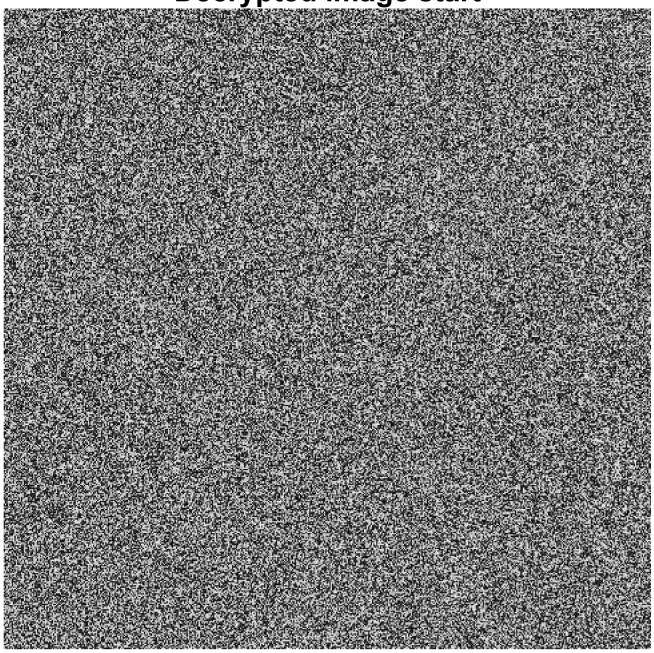

```
imshow(encryptedimage);
title('Final Encrypted image');
```

Final Encrypted image



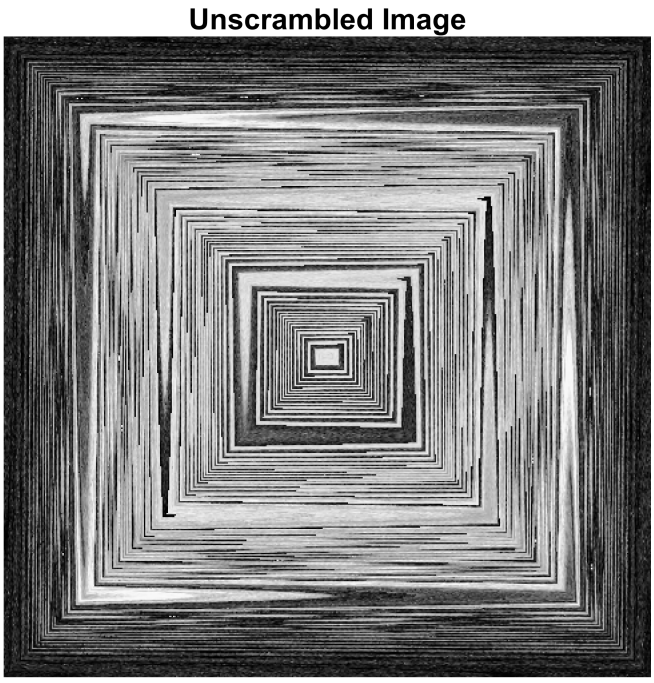
```
imshow(decryptedimage);
title('Decrypted image start');
```

Decrypted image start



```
imshow(finalconstructed)
```

```
title('Unscrambled Image');
```




```
imshow(reconstructed_image);
title('Reconstructed Grayscale Image');
```