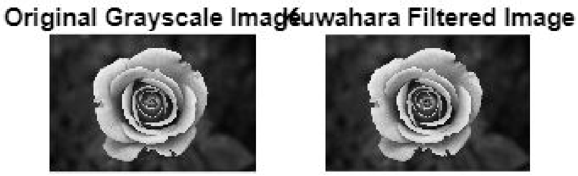
```
% Kuwahara filter is a non linear smoothing filter that used in edge preserving
smoothing. While traditional filters like Gaussian or average filters blur edges,
the Kuwahara filter leaves them untouched but smooths other regions. The filter
operated by splitting up an area of approximation 5x5 or 7x7 into four overlapping
quadrants. We then calculate the mean and variance (effectively how much change
in pixel value there is in that region) of each subregion. It then selects the
sub-region with the smallest variance of n*n data pixels from that specific patch
(the smoothest sub region), and replaces the centre pixel of that n*n window by
average value of all pixels within this sub-region. This approach keeps edges and
not changes pixel values where the variance is low (because areas with significant
variance are where we find edges most of the times)

% first step converting image
img = imread('input1.jpg');
if size(img, 3) == 3
    gray_img = rgb2gray(img);
else
    gray_img = img;
end

% Step 2: Apply Kuwahara filter
filtered_img = kuwahara_filter(gray_img, 5);  % Use a 5x5 window

% Step 3: Display the original and filtered images side by side
figure;
subplot(1, 2, 1);
imshow(gray_img);
title('Original Grayscale Image');

subplot(1, 2, 2);
imshow(filtered_img);
title('Kuwahara Filtered Image');
```



```
% Kuwahara filter function implementation
function output_img = kuwahara_filter(input_img, window_size)
```

```
    [rows, cols] = size(input_img);
    half_window = floor(window_size / 2);

    % Pad the image to handle the border pixels
    padded_img = padarray(input_img, [half_window half_window], 'symmetric');

    output_img = zeros(rows, cols, 'like', input_img);

    for i = 1:rows
        for j = 1:cols
            % Extract 4 overlapping subregions within the window
            region1 = padded_img(i:i+half_window, j:j+half_window);
            region2 = padded_img(i:i+half_window, j+half_window:j+window_size-1);
            regio3 = padded_img(i+half_window:i+window_size-1, j:j+half_window);
            regin4 = padded_img(i + half_window : i + window_size - 1 , j +
half_window : j + window_size - 1 );

            % Compute mean and variance for each subregion
            mean1 = mean(region1(:)); var1 = var(double(region1(:)));
            mean2 = mean(region2(:)); var2 = var(double(region2(:)));
            mean3 = mean(regio3(:)); var3 = var(double(regio3(:)));
            mean4 = mean(regin4(:)); var4 = var(double(region4(:)));

            % Select the subregion with the minimum variance
            [~, min_index] = min([var1, var2, var3, var4]);
            switch min_index
                case 1
                    output_img(i, j) = mean1;
                case 2
                    output_img(i, j) = mean2;
                case 3
                    output_img(i, j) = mean3;
                case 4
                    output_img(i, j) = mean4;
            end
        end
    end
end
```