```
%The Kuwahara filter is a non-linear smoothing filter used in image processing,
% which preserves edges while reducing noise. It divides the local area around a
pixel into overlapping regions and computes the mean and variance in each region.
% The filter then selects the region with the lowest variance and replaces the
center pixel with the mean value of that region.
% This helps in preserving edges better compared to other smoothing filters.

% Loading the image from the specified path and converting it to grayscale
input_image = imread('C:\Users\USER\Pictures\BQPE6282.JPG');
grayscale_image = rgb2gray(input_image);
grayscale_image = double(grayscale_image);  % Converting to double for precision

% Kuwahara filter implementation
function output_image = kuwahara_filter(img, window_size)
    % Pad the image to handle borders
    half_window = floor(window_size / 2);
    padded_image = padarray(img, [half_window, half_window], 'symmetric');

    % Get the dimensions of the original image
    [rows, cols] = size(img);
    output_image = zeros(rows, cols);

    % Iterate through every pixel in the original image
    for i = 1:rows
        for j = 1:cols
            % Define the window coordinates, ensuring they don't exceed image bounds
            top = i;
            bottom = i + window_size - 1;
            left = j;
            right = j + window_size - 1;

            % Extract the 4 overlapping regions
            region1 = padded_image(top:top+half_window,
left:left+half_window);          % Top-left
            region2 = padded_image(top:top+half_window,
left+half_window+1:right);       % Top-right
            region3 = padded_image(top+half_window+1:bottom,
left:left+half_window);     % Bottom-left
            region4 = padded_image(top+half_window+1:bottom,
left+half_window+1:right);  % Bottom-right

            % Compute the means and variances of each region
            mean1 = mean(region1(:)); variance1 = var(region1(:));
            mean2 = mean(region2(:)); variance2 = var(region2(:));
            mean3 = mean(region3(:)); variance3 = var(region3(:));
            mean4 = mean(region4(:)); variance4 = var(region4(:));

            % Find the region with the minimum variance
            [~, min_region] = min([variance1, variance2, variance3, variance4]);
```

```
            % Assign the mean of the region with the smallest variance to the
output pixel
            switch min_region
                case 1
                    output_image(i, j) = mean1;
                case 2
                    output_image(i, j) = mean2;
                case 3
                    output_image(i, j) = mean3;
                case 4
                    output_image(i, j) = mean4;
            end
        end
    end
    output_image = uint8(output_image);  % Converting the output to uint8
end

% Applying the Kuwahara filter with a window size of 5x5
filtered_image = kuwahara_filter(grayscale_image, 5);

% Saving and displaying the filtered image
imwrite(filtered_image, 'C:\Users\USER\Desktop\kuwahara_filtered.png');
figure, imshow(filtered_image), title('Kuwahara Filtered Image');
```

Kuwahara Filtered Image