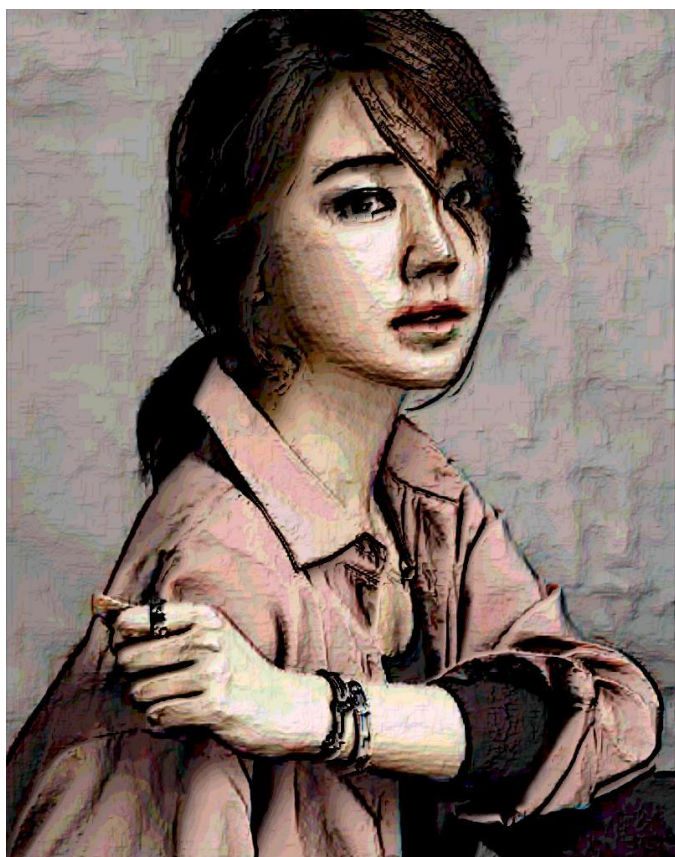


## Cartoon Effect!!



### STEP 1: Posterization (Image Quantization) :

```
threshRGB = multithresh(im,N); %thresh is a 1xN vector which can be used to convert
                                % image im into an image with N + 1 discrete levels.
value = [0 threshRGB(2:end) 255];
quantRGB = imquantize(im, threshRGB, value);
```

### STEP 2: Image Smoothing

### STEP 3: Edge Emphasizing

| MATLAB functions                                   | Example  |
|--|--|
| <code>B = <b>imfilter</b>(I, h)</code>             | Filter the image <code>A</code> with the filter <code>h</code> . Create a filter <code>h</code> using <code>fspecial()</code>  |
| <code>h = <b>fspecial</b>(type, parameters)</code> | <p>Creates a two-dimensional filter <code>h</code> of the specified <code>type</code>.</p> <ul style="list-style-type: none"><li><code>h = fspecial('average', hsize)</code> returns an averaging filter <code>h</code> of size <code>hsize</code>. The argument <code>hsize</code> can be a vector specifying the number of rows and columns in <code>h</code>, or it can be a scalar, in which case <code>h</code> is a square matrix. The default value for <code>hsize</code> is <code>[3 3]</code>.</li><li><code>h = fspecial('gaussian', hsize, sigma)</code> returns a rotationally symmetric Gaussian lowpass filter of size <code>hsize</code> with standard deviation <code>sigma</code> (positive). <code>hsize</code> can be a vector specifying the number of rows and columns in <code>h</code>, or it can be a scalar, in which case <code>h</code> is a square matrix. The default value for <code>hsize</code> is <code>[3 3]</code>; the default value for <code>sigma</code> is 0.5.</li><li><code>h = fspecial('prewitt')</code> returns the 3-by-3 filter <code>h</code> (shown below) that emphasizes horizontal edges by approximating a vertical gradient. If you need to emphasize vertical edges, transpose the filter <code>h'</code>.<br/><math display="block">\begin{bmatrix} 1 &amp; 1 &amp; 1 \\ 0 &amp; 0 &amp; 0 \\ -1 &amp; -1 &amp; -1 \end{bmatrix}</math><p>To find vertical edges, or for x-derivatives, use <code>h'</code>.</p></li><li><code>h = fspecial('sobel')</code> returns a 3-by-3 filter <code>h</code> (shown below) that emphasizes horizontal edges using the smoothing effect by approximating a vertical gradient. If you need to emphasize vertical edges, transpose the filter <code>h'</code>.<br/><math display="block">\begin{bmatrix} 1 &amp; 2 &amp; 1 \\ 0 &amp; 0 &amp; 0 \\ -1 &amp; -2 &amp; -1 \end{bmatrix}</math></li></ul> |