Lab1 - Calculating and Plotting Image Histogram

This lab aims at learning how to scan ***graylevel*** and ***RGB*** images to calculate and plot a useful image statistic called ***histogram*** . The histogram of a graylevel image is a vector that stores for each color the number of pixels of that particular color. The size of the vector is equal to the number of gray colors (256 for uint8).

This lab consists of an **assignment** and **tutorial steps** that help to get started with the assignment.

Contents

Step 1: IPT function imhist

Image Processing Toolbox in Matlab has a function for calculating image histogram. The following code shows it in action.

```
I = imread('graypeppers.png');
imshow(I);
```

```
h = imhist(I); % calculate image histogram
size(h) % show size of h vector
```

```
ans =
```

```
    256     1
```

```
h(100) % number of pixels with color equal to 100
```

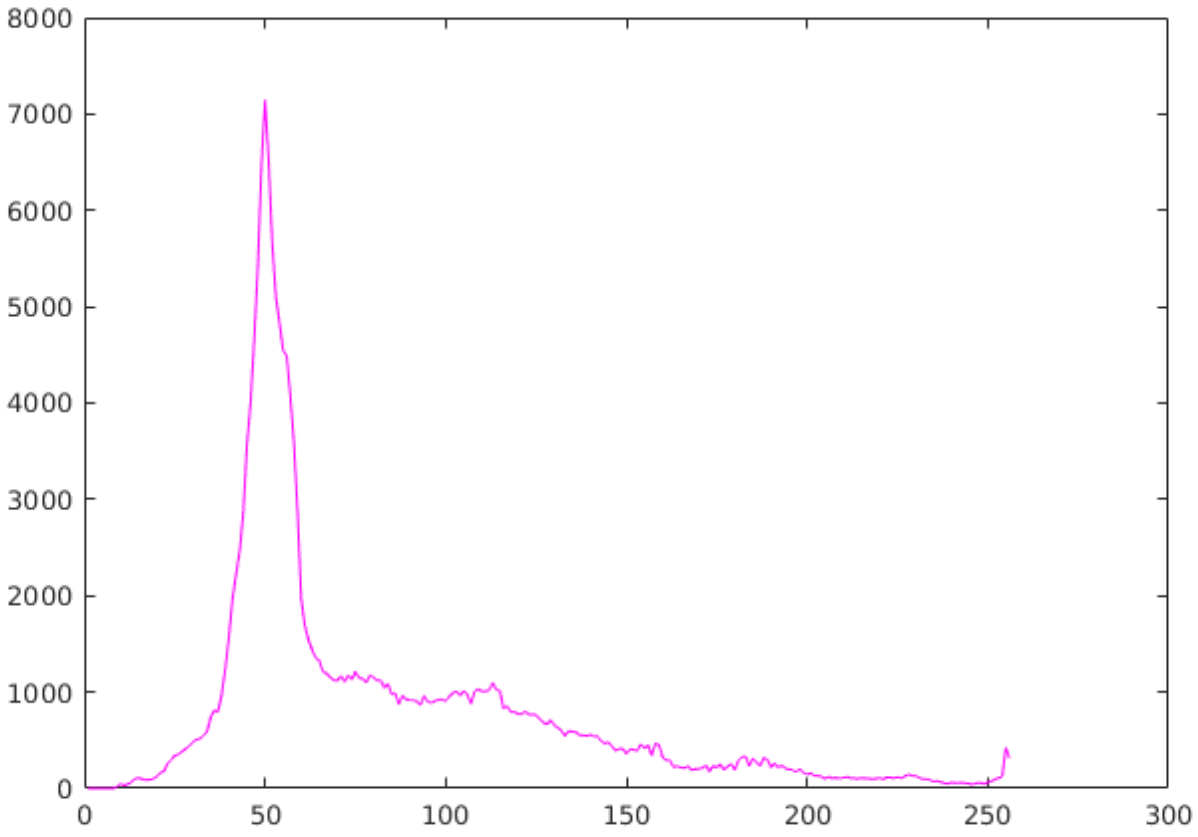
```
ans =

    907
```

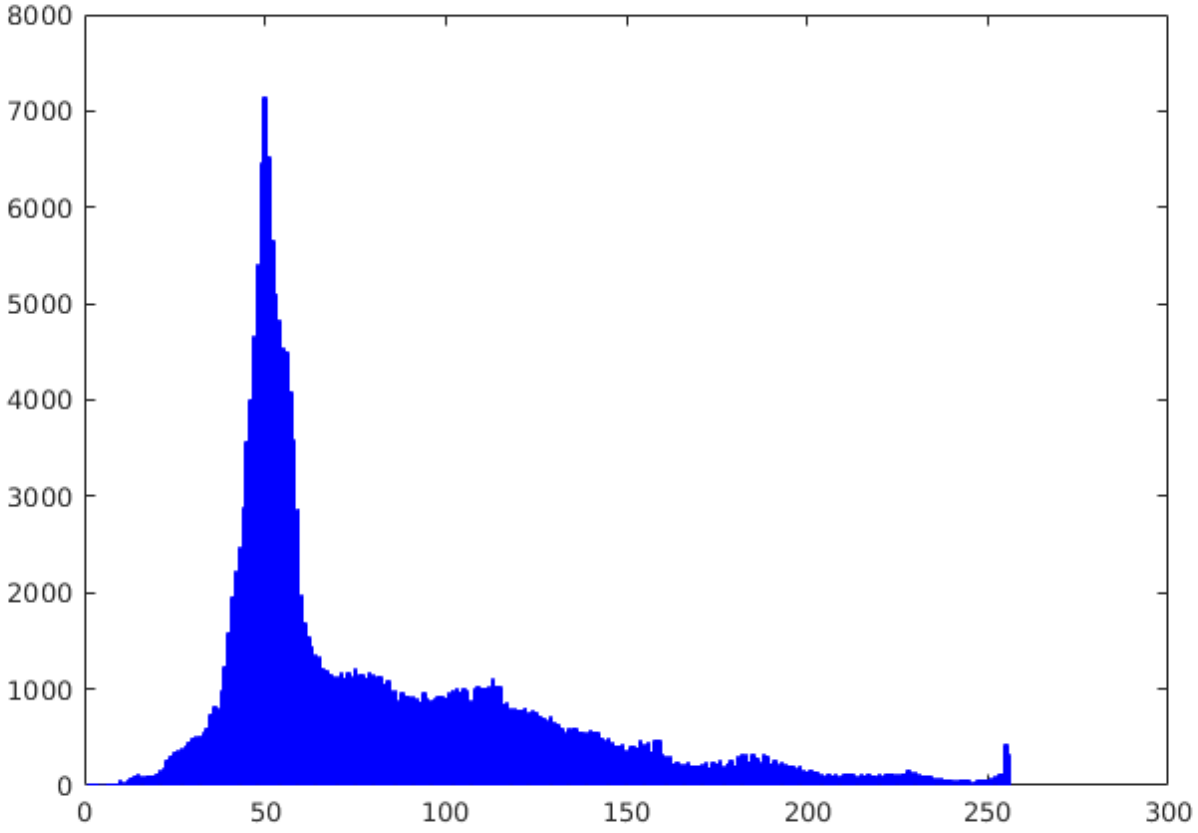
Step 2: Plotting histogram

You can plot image historgram as well using **plot**, **bar** or **stem** functions

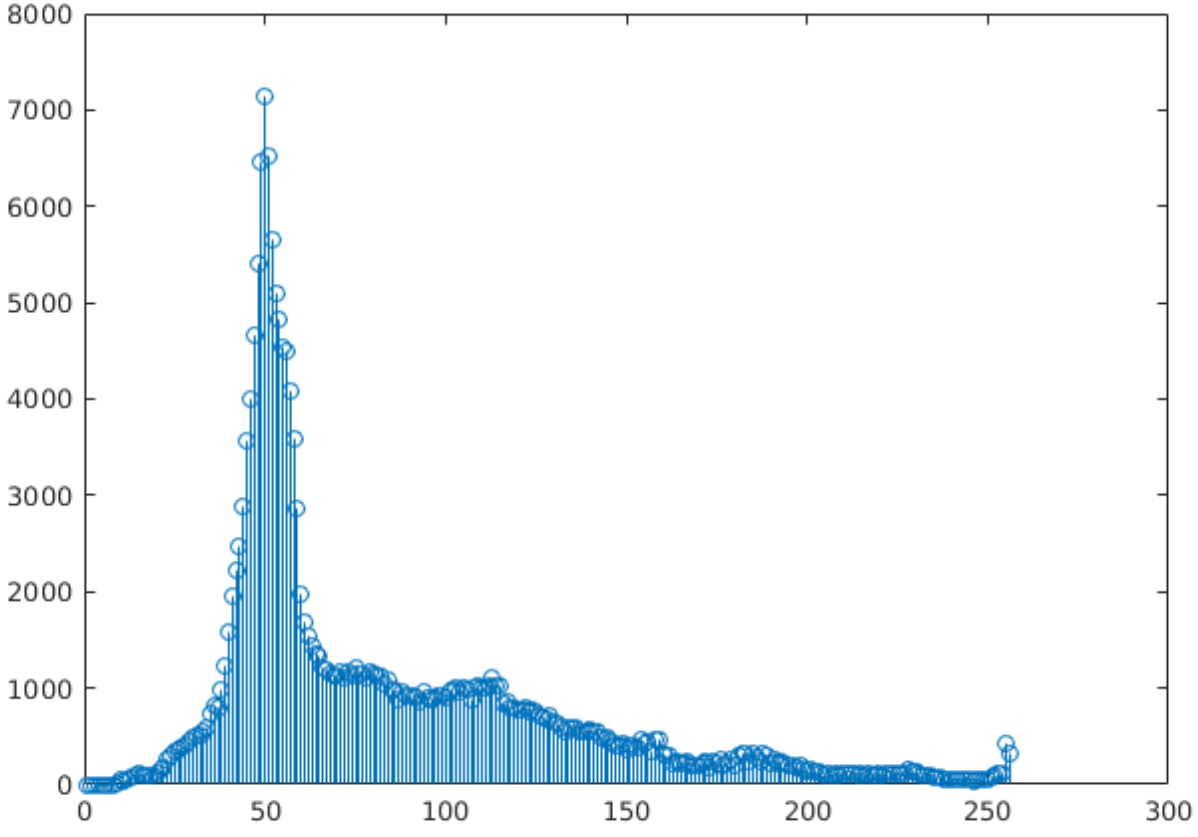
```
plot(h,'magenta');
```



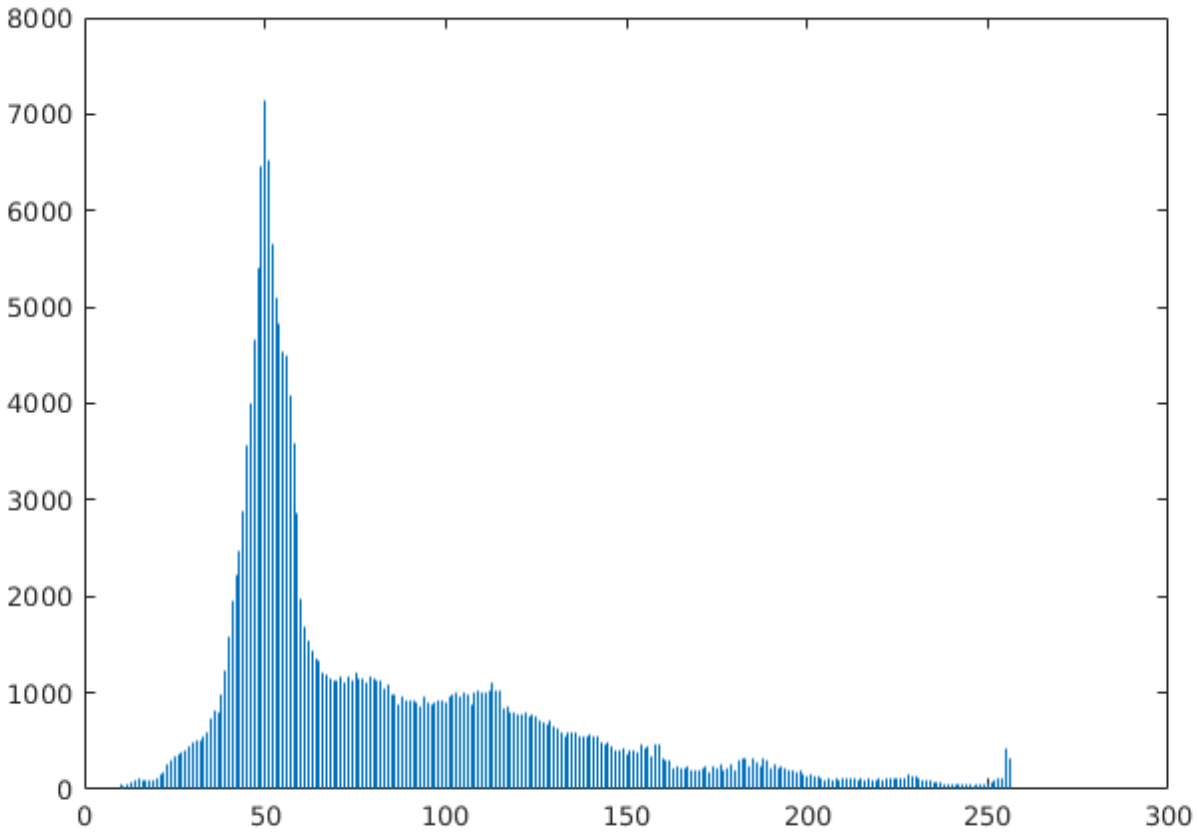

```
bar(h,'blue');
```


8000
7000
6000
5000
4000
3000
2000
1000
0
0
50
100
150
200
250
300

```
stem(h);
```

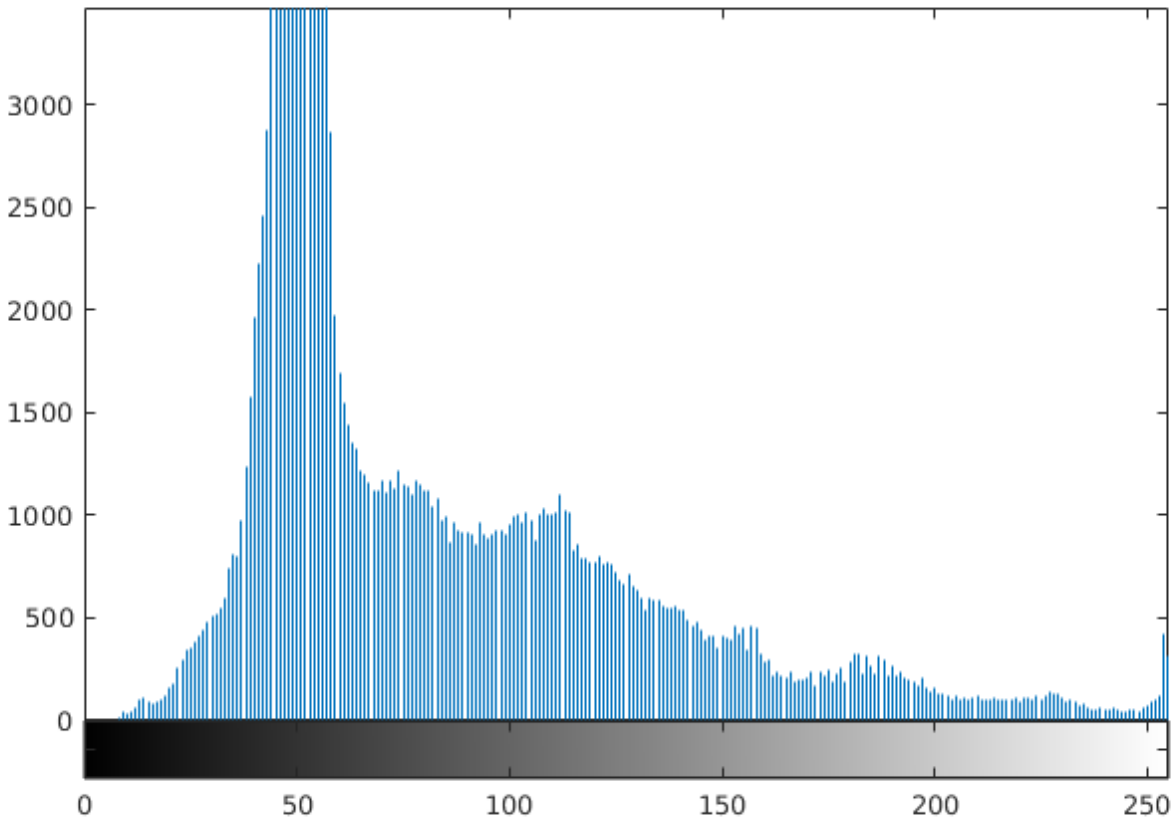

8000
7000
6000
5000
4000
3000
2000
1000
0
0
50
100
150
200
250
300

```
stem(h,'Marker','none');
```



imhist can also be used directly to visualize image histogram

```
imhist(I);
```



Step 3: Our function xist

Because we are cool dudes (or peppers) we write our own histogram calculating function **xist**:

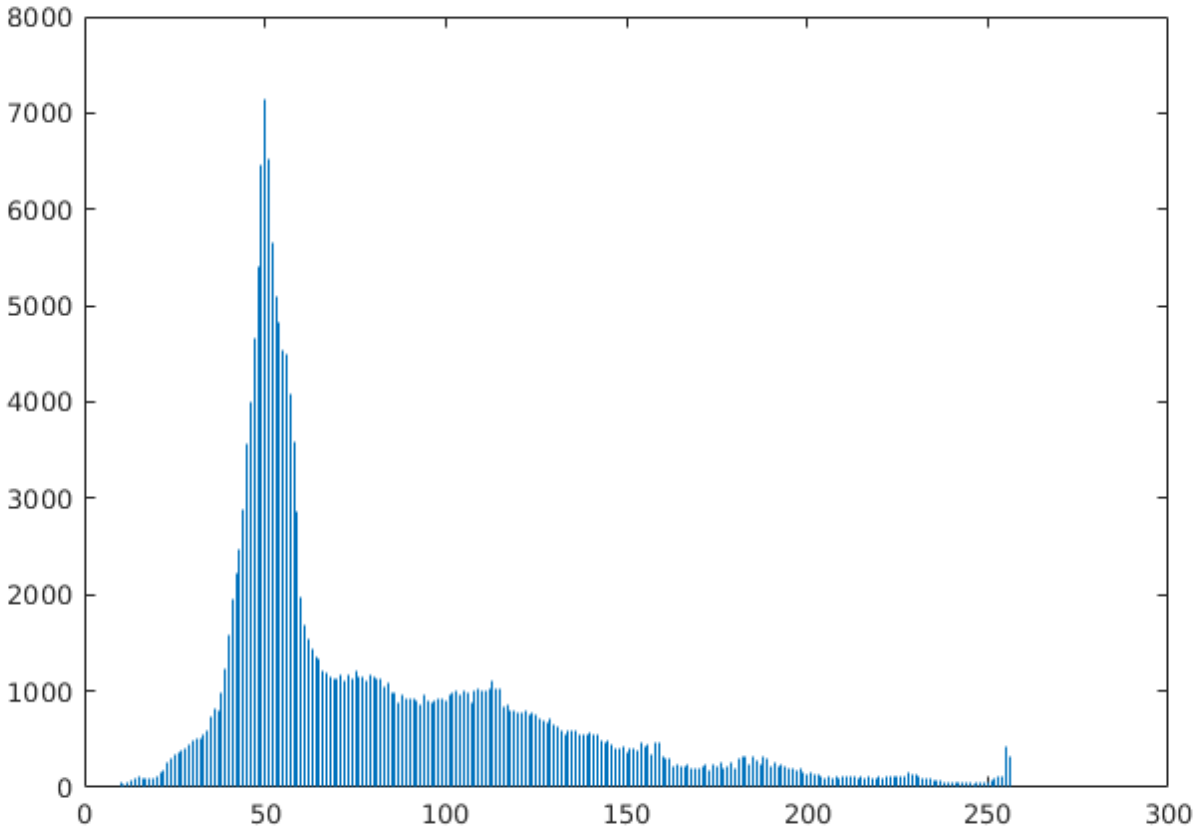
```
function h = xist(I)
% XIST Calculate histogram of graylevel uint8 image I

    h = zeros(1,256); % creating array of 256 zeros
    [height,width]=size(I); % getting dimensions of I
    % counting colors
    for i=1:height
        for j=1:width
           color = I(i,j);
           h(color+1) = h(color+1) + 1;
        end
    end

end
```

and here is **xist** function in action:

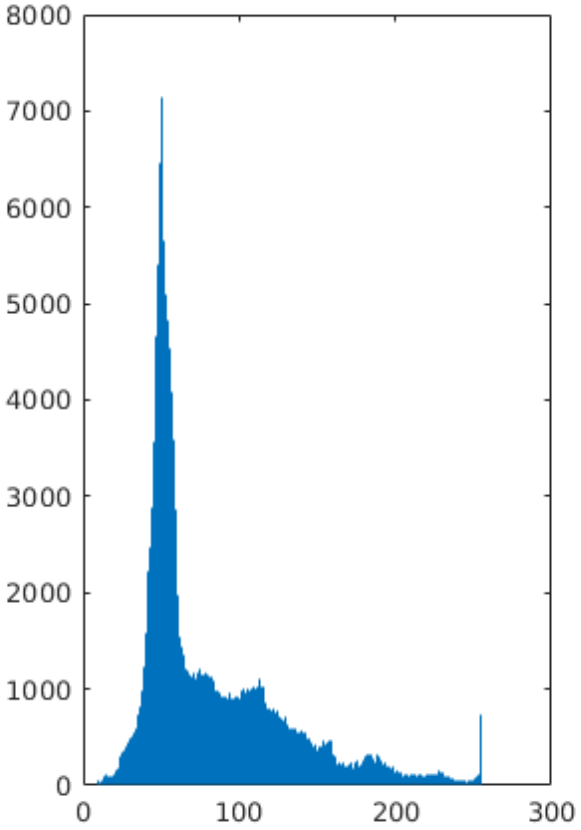
```
x = xist(I);
stem(h,'Marker','none');
```



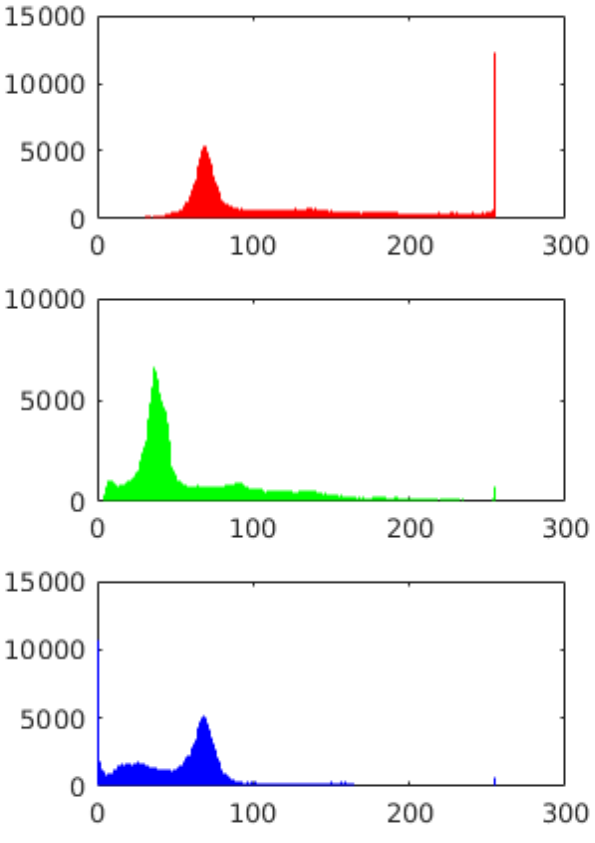
Assignment

As you might have noticed the IPT's **imhist** function has a small bug. So write your own function **imxist** that calculates image histogram (perhaps using **xist** function) and plots it with the image itself. In case the image is colorful, i.e. containing red, green and blue components, make your function plot histograms of each components independently as if they were really independent (in fact they are not).

```
G = imread('graypeppers.png');
RGB = imread('peppers.png');
imxist(G);
```


8000
7000
6000
5000
4000
3000
2000
1000
0
0
100
200
300

```
imxist(RGB);
```


15000
10000
5000
0
0
100
200
300
10000
5000
0
0
100
200
300
15000
10000
5000
0
0
100
200
300

Follow the below steps to submit your assignment

1. Write **imxist** function and check if it works correctly
2. Populate the stub that is shown below (you can find it in **Lab1stub.m**) with correct data and code
3. Publish the stub as *html*
4. Open the published html file in Chromium (or Google-chrome) browser and print it as *pdf* (in print dialogue uncheck ***Header and Footers*** option and check ***Background graphics*** option for nicer output)
5. Submit the resultant pdf and wait for me to grade it

The stub that you should use to make my life easier...

```
%% Lab1 Assignment
% Name: *Andrey*
% Surname: *Godgivenson*
% Group: *EN4-Z-01*
% Your comments...
%% Question 1
% What a heck is wrong with *imhist*?
% Well, it's color is strange...
%% imxist code
%
% <include> imxist.m </include>
%
%% Demo1 : Your Title for Demo1
% Your comments for Demo1
I = imread('peppers.png');
imxist(I);
%% Demo2 : Your Title for Demo2
% Your comments for Demo2
your_code_for_Demo2
%% DemoN : Your Title for DemoN
% Your comments for DemoN
your_code_for_DemoN
```

GOOD LUCK!!!

Published with MATLAB® R2015b