```
%-------------------------------------------------------------%
%-------- Matlab Image Intensity Levels converter -----------%
%-------------------------------------------------------------%
%
%  !!!! TIF version !!!!
%
% V0.11 2004/12/16
% Changelog:
% from V0.1 (2004/12/15): added +0.5 for correct rounding
%
%
% Description:
% This Matlab script can be used to convert a 16-bit tif image into a 8bit**
% tif* (100% quality) by defining the intensity range equivalent to 0..255
%  (*: the output format can be changed into any other desired format, e.g. tif)
%  (**: the bit level may be adjusted too if desired - possible with tif images)
%
% Stage: Hack level, quick solution
%
% Input arguments:
%       1st: "path" - path string to the folder where the images to be converted
%                     are contained (e.g. 'c:\pivimg\')
%       2nd: "Imin" - original level equivalent to 0
%       3rd: "Imax" - original level equivalent to 255
%
% example to convert the original levels ranged 100 to 300 into 0 to 255
% (linearly interpolated / shifted) with all files in folder 'd:\myimages\'
% type: LevelConv('d:\myimages\', 100, 300)
%
%
% basic function description (based on the assumption of a 16 bit tif originating
% from a 10 bit camera and shifted by multiplication with 2^6)
% note images aquired with a different software than epix might behave differently
% 1. the 16-bit tif is read
% 2. the 16-bit tif is shifted to 10 bit by multiplication with 2^-6
% 3. the lower level is set to zero by subtracting the lower level (e.g. 100)
% 4. the range of 0-255 is achieved by multiplication (linear shifting)


function LevelConv(path, Imin, Imax)     % function definition

% Adjustable parameters
imgReadFormat = 'tif';  % format of image to read, choose from: 'tif', 'jpg', ...
imgWriteFormat = 'tif'; % format to write the image to; carefull: set quality etc
                        % as well!
PicResBit = 16;         % picture resolution
CamResBit = 10;         % camera resolution in bit

% get files from path
 files = dir ([path, '*.', imgReadFormat]);

% process all files in the path
for i=1:size(files,1)
   X = imread([path,files(i).name]); % read a particular image file
   ImgFName = [path, '8bit', ...
       files(i).name(1:findstr(files(i).name,imgReadFormat)-1), imgWriteFormat]
% image file name to be written display not suppressed to allow viewing progress
   X = uint8( ( double(X)*2^(CamResBit-PicResBit) - Imin ) ...
                /(Imax-Imin)*255 + 0.5); % image manipulation as described
       % note any levels above 255 will be cropped to 255 as unlike other software
       % Matlab won't rebegin at zero due to bit shifting (makes life easier)
   imwrite(X,ImgFName,imgWriteFormat); % writing the new image
end

```