THIS IS A MATLAB CODE SAMPLE WHICH I WROTE IN MY IMAGE PROCESSING CLASS.

```
% Experiment 1

I= imread('sunset.jpg')
figure, image(I), truesize; title('A truecolor image')
xlabel('Picture is taken from http://xinature.com/flower-flowers-italy-field-beautiful-clouds-mountains-spring-blue-red-sunset-green-nature-desktop-wallpaper/')
[R C B]= size(I)
class(I)
r= I(:,:,1); % extracting red band image
g= I(:,:,2); % extracting green band image
b= I(:,:,3); % extracting blue band image
% Applying the gamma correction function to decrease and increase the
% brightness of each r g b band images:
r1= correctgama (r, 2); % Increasing gamma of r band by a factor of 2
R1= cat (3, r1, g, b); % recombining the altered r band and original g and b bands.
%figure, image(R1), truesize; title('gamma of red band increased  by a factor of 2')
R1_cut= R1(100:800, 600:1000, :);% cropping the image
figure, image(R1_cut), truesize; title('gamma of red band increased  by a factor of 2')
r2= correctgama (r, 0.25); % decreasing gamma of r band by a factor of 0.5
R2= cat (3, r2, g, b);% recombining the altered r band and original g and b bands.
%figure, image(R2), truesize; title('gamma of red band decreased  by a factor of 0.5')
R2_cut= R2(100:800, 600:1000, :); % cropping the image
figure, image(R2_cut), truesize; title('gamma of red band decreased  by a factor of 0.25')
g1= correctgama (g, 2); % Increasing gamma by a factor of 2
G1= cat (3, r, g1, b);% recombining the altered g band and original r and b bands.
%figure, image(G1), truesize; title('gamma of green band increased  by a factor of 2')
G1_cut= G1(100:800, 600:1000, :);% cropping the image
figure, image(G1_cut), truesize; title('gamma of green band increased  by a factor of 2')
g2= correctgama (g, 0.25); % decreasing gamma by a factor of 0.5
G2= cat (3, r, g2, b);% recombining the altered g band and original r and b bands.
%figure, image(G2), truesize; title('gamma of green band decreased  by a factor of 0.5')
G2_cut= G2(100:800, 600:1000, :);% cropping the image
```

```
figure, image(G2_cut), truesize; title('gamma of green band decreased  by a
factor of 0.25')
b1= correctgama (b, 2); % Increasing gamma by a factor of 2
B1= cat (3, r, g, b1);% recombining the altered b band and original g and r
bands.
%figure, image(B1), truesize; title('gamma of blue band increased  by a factor of
2')
B1_cut= B1(100:800, 600:1000, :);% cropping the image
figure, image(B1_cut), truesize; title('gamma of blue band increased  by a factor
of 2')
b2= correctgama (b, 0.25); % decreasing gamma by a factor of 0.5
B2= cat (3, r, g, b2);% recombining the altered b band and original g and r
bands.
%figure, image(B2), truesize; title('gamma of blue band decreased  by a factor of
0.5')
B2_cut= B2(100:800, 600:1000, :);
figure, image(B2_cut), truesize; title('gamma of blue band decreased  by a factor
of 0.25')
```

EXPERIMENT 2:

```
% Experiment 2(a)
I= imread('sunset.jpg');
[h, s, v]= RGBtoHSV(I); % converting image I from rgb to hsv by calling the
function
v=uint8(v);% class of value image v to uint8
% Applying the gamma correction function on the value image to increase and
% decrease it:
v1= correctgama (v, 2); % Increasing gamma by a factor of 2
rgb_v1= hsvTOrgb(h,s, v1);% convering the image back to rgb with altered v
figure, image(rgb_v1), truesize;title('Reconverted r g b image with increased
gamma of v band')
v2= correctgama (v, 0.25); % Increasing gamma by a factor of 2
rgb_v2= hsvTOrgb(h,s, v2);% convering the image back to rgb with altered v
figure, image(rgb_v2), truesize;title('Reconverted r g b image with decreased
gamma of v band')
```

```
% Experiment 2(b)
% linearly increasing and decreasing the saturation:
I= imread('sunset.jpg');
[h, s, v]= RGBtoHSV(I);
s_up=2; %increasing s by factor of 2
S_up=s_up*s;
S_up(S_up>1)=1; % clipping the
rgb_s_up= hsvTOrgb(h,S_up, v); % converting the kimage back to rgb with altered s
figure, image (rgb_s_up), truesize; title ('Reconstructed rgb image having
saturation increased');
```

```
s_down=0.25;%decreasing s by factor of 2
S_down=s_down*s;
S_down(S_down>1)=1;
rgb_s_down= hsvTOrgb(h,S_down, v);% converting the kimage back to rgb with
altered s
figure, image (rgb_s_down), truesize; title ('Reconstructed rgb image having
saturation decreased');
```

```
%Experiment 2(c)
% Applying a constant circular shift to the H band to shift the colors in the
green and blue
% direction:
mod(h,2*pi) ; % mod is used to wrap around the pixels so that it does not exceed
2pi
H1= h+ (2*pi/3); % for the green direction
hsm1= mod (H1, 2*pi);% hue shifted towards blue
mod(h,2*pi) ;
H2= h+ (4*pi/3); % for the blue direction
hsm2= mod (H2, 2*pi); % hue shifted towards blue
rgb_hsm1= hsvTOrgb(hsm1,s, v); % converting the image back to rgb with hue
shifted towards green
figure, image(rgb_hsm1), truesize,title('Reconstructed rgb image having hue
shifted towards green');
rgb_hsm2= hsvTOrgb(hsm2,s, v);% converting the image back to rgb with hue shifted
towards green
figure, image(rgb_hsm2), truesize;title('Reconstructed rgb image having hue
shifted towards blue');
```

EXPERIMENT 3:

```
%Experiment 3
% Correction of an image with a hue shift:
I= imread('LAB 3\PhotoDiskTestImageHueShifted.bmp');
figure, image(I), truesize;
A=I(329, 165,:); % sampling the pixel in the should-be-blue sqaure
[h, s, v]= RGBtoHSV(A);% converting the pixel to HSV
X= zeros(3,360)
for x= 1:360 % formiong a loop for all angles from 0 to 360 degrees.
 x1= x*(pi/180);
   H= h+ (x1); % shifts the hue in a particular direction depending on the value
of x
   h1= mod (H, 2*pi);
   rgb= hsvTOrgb(h1,s, v); % converting the image back to rgb with shifted hue.
r= A(:,:,1); % extracting red band image
g= A(:,:,2); % extracting green band image
b= A(:,:,3); % extracting blue band image
   X(1,x)=r
```

```
   X(2,x)=g
   X(3,x)=b
b=rgb(:,:,3)% b is maximum for x= 304. The value of b at x=304 is 149.
x
end
[h, s, v]= RGBtoHSV(I);
 H= h+ (304*pi/180); %shifting the hue for angle x=304*pi/180 radians
 h1= mod (H, 2*pi);
rgb= hsvTOrgb(h1,s, v)
figure, image(rgb), truesize; title ('Image in its original form');
```

EXPERIMENT 4:

```
% Experiment 4
I=imread('AHS_Hotel_Comicon_667x1000_discolored.bmp');
imshow(I)
rr=size(I,1);
cc=size(I,2);
bb=size(I,3);
[c,r]=ginput;   % Select pixels.End with ENTER
r=round(r);
c=round(c);
n=size(r,1);
X=zeros(3,n);
for i=1:n
    X(1,i)=double(I(r(i),c(i),1));
    X(2,i)=double(I(r(i),c(i),2));
    X(3,i)=double(I(r(i),c(i),3));
end
Y= [255 51 255 255 38 255; 242 0 210 51 125 255; 40 25 235 51 113 0 ];% Actual
color matrix
X=double(X);
Y=double(Y);
A=zeros(3,3);
A=X*X'
Z=inv (A)
B= X'*Z; % transformation matrix B
B=Y*B
[R C B]=size(I)
J = reshape(((B*(reshape(double(I),R*C,3))')'),R,C,3);
m = min(J(:));
M = max(J(:));
J = uint8(255*(J-m)/(M-m));
figure, image(J), truesize; title('restored image')
p=pdf(J)
```

```
plot(p)
```

EXPERIMENT 5:

```
% Experiment 5:
%Checking that the functions are running without an error:
I= imread('lighthouse.bmp');
figure, image(I), truesize;

M= [0.5767 0.1856 0.1882 ; 0.2974 0.6273 0.0753 ; 0.0270 0.0707 0.9911];% M is a
3X3
[X,Y,Z] = RGBtoXYZ(I, M)

v=[94.811 100 107.304]
[L,a,b] = XYZtoLab( X, Y, Z, v )

[X,Y,Z] = LabtoXYZ( L, a, b, v )

M= [2.0414 -0.5694 -0.3447; -0.9693 1.8760 0.0416; 0.0134 -0.1184 1.0154];
[R,G,B] = XYZtoRGB(X, Y, Z, M)
 R=uint8(R);
 G=uint8(G);
 B=uint8(B);
 C= cat(3,R,G,B)
 figure, image(C),truesize; title('original image obtained')% Thus all the
functions are running properly


% Converting the L a b, by altering the L and keeping a and b same, to XYZ and
then XYZ to
% rgb:
% performing multiplication operation on L:
 L_new_1= L*(1.5) % L increased 1.5 times
[X,Y,Z] = LabtoXYZ( L_new_1, a, b, v ) %calling Lab to XYZ for altered L
M= [2.0414 -0.5694 -0.3447; -0.9693 1.8760 0.0416; 0.0134 -0.1184 1.0154];%matrix
for XYZ to RGB (adobe)
[R,G,B] = XYZtoRGB(X, Y, Z, M) % calling XYZ to RGB
 R=uint8(R); %converting R G B to uint8
 G=uint8(G);
 B=uint8(B);
RGB_new_L1= cat(3, R,G,B) %concatenating RGB
figure, image(RGB_new_L1), truesize; title('image with L increased 1.5 times')
% performing addition operation on L:
L_new_2= L+20 % L increased by 20
[X,Y,Z] = LabtoXYZ( L_new_2, a, b, v ) %calling Lab to XYZ for altered L
M= [2.0414 -0.5694 -0.3447; -0.9693 1.8760 0.0416; 0.0134 -0.1184 1.0154];
[R,G,B] = XYZtoRGB(X, Y, Z, M)
 R=uint8(R);
```

```
 G=uint8(G);
 B=uint8(B);
RGB_new_L2= cat(3, R,G,B)
figure, image(RGB_new_L2), truesize;title('image with L increased by 20')

% Converting the L a b, by altering the a and keeping L and b same, to XYZ and
then XYZ to
% rgb:
% performing multiplication operation on a:
a_new_1= a*5 % a increased by 5 times
[X,Y,Z] = LabtoXYZ( L, a_new_1, b, v ) %calling Lab to XYZ for altered a
M= [2.0414 -0.5694 -0.3447; -0.9693 1.8760 0.0416; 0.0134 -0.1184 1.0154];
[R,G,B] = XYZtoRGB(X, Y, Z, M)
R=uint8(R);
 G=uint8(G);
 B=uint8(B);
RGB_new_a1= cat(3, R,G,B)
figure, image(RGB_new_a1), truesize;title('image with a increased by 5 times')
% performing addition operation on a:
a_new_2= a+50 % a increased by 50
[X,Y,Z] = LabtoXYZ( L, a_new_2, b, v )%calling Lab to XYZ for altered a
M= [2.0414 -0.5694 -0.3447; -0.9693 1.8760 0.0416; 0.0134 -0.1184 1.0154];
[R,G,B] = XYZtoRGB(X, Y, Z, M)
R=uint8(R);
 G=uint8(G);
 B=uint8(B);
RGB_new_a2= cat(3, R,G,B)
figure, image(RGB_new_a2), truesize;title('image with a increased by 50')

% Converting the L a b, by altering the b and keeping a and L same, to XYZ and
then XYZ to
% rgb:
% performing multiplication operation on b:
b_new_1= b*10
[X,Y,Z] = LabtoXYZ( L, a, b_new_1, v )%calling Lab to XYZ for altered b
M= [2.0414 -0.5694 -0.3447; -0.9693 1.8760 0.0416; 0.0134 -0.1184 1.0154];
[R,G,B] = XYZtoRGB(X, Y, Z, M)
R=uint8(R);
 G=uint8(G);
 B=uint8(B);
RGB_new_b1= cat(3, R,G,B)
figure, image(RGB_new_b1), truesize;title('image with b increased 10 times')
% performing addition operation on b:
b_new_2= b+50
[X,Y,Z] = LabtoXYZ( L, a, b_new_2, v )%calling Lab to XYZ for altered b
M= [2.0414 -0.5694 -0.3447; -0.9693 1.8760 0.0416; 0.0134 -0.1184 1.0154];
[R,G,B] = XYZtoRGB(X, Y, Z, M)
R=uint8(R);
 G=uint8(G);
 B=uint8(B);
RGB_new_b2= cat(3, R,G,B)
```

```
figure, image(RGB_new_b2), truesize;title('image with b increased by 50')
```

ALL THE FUNCTIONS:

```
%FUNCTIONS:

function f= remap (I,LUT) % Matlab function to remap a uint8 image through a
look-up table (LUT):
f= LUT(I+1);
f= uint8(f);
end
% Matlab function to generate a LUT for for gamma adjustment of uint8
% images:
function G= gama_adjust(x)
LUT=(0:255);
G= 255*(LUT/255).^(1/x) ;
end
% function for gamma correction:
function Y= correctgama(I, gamma)
G=gama_adjust(gamma);
Y= remap(I,G);
end
function [h, s, v]= RGBtoHSV(I)
I=double(I)
r=I(:, :, 1); %red band of I extracted
g=I(:, :, 2); %green band of I extracted
b=I(:, :, 3);%blue band of I extracted
v=(r+g+b)/3;% value image of I
class(v) %checking class of v
V= cat(3,v, v, v);%Compute vector value image
S=I-V; %Compute vector saturation image
s= sum(S.^2, 3); %Compute scalar saturation image
s= sqrt (s); %Compute scalar saturation image
X= cat (3, 2*v, -v, -v);%Compute red axis vector image
x= sum(X.^2, 3);% Compute red axis scalar image
x= sqrt(x);% Compute red axis scalar image
s(s==0)= 1; % set 0 scalar sats to 1s (gray pixels)
x(x==0)= 1;% set 0 length red-vecs to 1s. (black pixels)
c= sum(S.*X, 3);% cos(hue) as dot product of vec sat & vec red
nc= c./(s.*x);% normalized c
nc(nc>1)= 1; % correct for round-off errors
nc(nc<-1)= -1; % correct for round-off errors
h= acos(nc); % hue is the inverse cosine
m= b>g;% Create logical image
h = ~m.*h +m.*(2.*pi - h);
s= s/208.2066 ; %Normalize the saturation
end
function rgb= hsvTOrgb(h,s, v)  % Function to convert HSV image to RGB image.
hc = h(:);%Reshape R×C image h into RC×1 column vectors
[R C]= size (h)
```

```
sc = s(:); %Reshape R×C image s into RC×1 column vectors
v=double(v)
vc = v(:); %Reshape R×C image v into RC×1 column vectors
sc=sc*208.2066; %Denormalize the saturation image
xc= sc.*cos(hc); %Compute the x-coordinate image
yc= sc.*sin(hc); %Compute the y-coordinate image
V= [vc vc vc].'; %Construct the value vector image
A= 0.408*[2 0; -1 1.732; -1 -1.732] %Construct the 1st two columns of rotation
matrix A
s= [xc yc].'; %Rotate the saturation image
S= A*s
I=S+V; %Add the result to V to recover image I(rgb)
I_uint8= uint8(I); %Convert image I to class uint8
rgb = reshape(I_uint8',[R C 3]); %reshape I(rgb) to R×C×3
end
function h= histogram(I)
h= histcounts (I, (0:256))';
class(h) %checking class of h
end
%Matlab function for computing probability density funtions of one-band
%images:
function p= pdf(r)
h= histogram(r); % calling the histogram function
p=h/length(r(:));
end
% function to convert RGB to XYZ:
function  [X,Y,Z] = RGBtoXYZ(I, M)
I = double(I);
R = I(:,:,1);
G = I(:,:,2);
B = I(:,:,3);
X = M(1,1).*R + M(1,2).*G + M(1,3).*B;
Y = M(2,1).*R + M(2,2).*G + M(2,3).*B;
Z = M(3,1).*R + M(3,2).*G + M(3,3).*B;
end
% function to convert XYZ to RGB:
function  [R,G,B] = XYZtoRGB(X, Y, Z, M)
R = M(1,1).*X + M(1,2).*Y + M(1,3).*Z;
G = M(2,1).*X + M(2,2).*Y + M(2,3).*Z;
B = M(3,1).*X + M(3,2).*Y + M(3,3).*Z;
end
% function to convert X, Y , and Z bands with one illuminant to X,
% Y , and Z bands with another illuminant using Bradford matrix:
function [X,Y,Z]= XYZtoXYZ (Xsrc,Ysrc, Zsrc, vsrc, vdst)
% v is the illuminant taken as a row vector.
Xsw= vsrc(1,1) % X for source white
Ysw=vsrc(1,2)% Y for source white
Zsw= vsrc(1,3)% Z for source white
Xdw=vdst(1,1)% X for destination white
Ydw=vdst(1,2)% Y for destination white
```

```
Zdw= vdst(1,3)% Y for destination white
Rdw= 0.8951.*Xdw + 0.2664.*Ydw + (-0.1614).*Zdw;
Gdw= (-0.7502).*Xdw + 1.7135.*Ydw + 0.0367.*Zdw;
Bdw= 0.0389.*Xdw + (-0.0685).*Ydw + 1.0296.*Zdw;
Rsw= 0.8951.*Xsw + 0.2664.*Ysw + (-0.1614).*Zsw;
Gsw= (-0.7502).*Xsw + 1.7135.*Ysw + 0.0367.*Zsw;
Bsw= 0.0389.*Xsw + (-0.0685).*Ysw + 1.0296.*Zsw;
% Bradford matrix BM:
BM= [0.9870 -0.1471 0.1600; 0.4323 0.5184 0.0493; -0.0085 0.04 0.9685]*[Rdw/Rsw 0
0; 0 Gdw/Gsw 0; 0 0 Bdw/Bsw]*[0.8951 0.2664 -0.1614; -0.7502 1.7135 0.0367;
0.0389 -0.0685 1.0296]
X= BM(1,1).*Xsrc + BM(1,2).*Ysrc+ BM(1,3).*Zsrc;
Y= BM(2,1).*Xsrc + BM(2,2).*Ysrc+ BM(2,3).*Zsrc;
Z= BM(3,1).*Xsrc + BM(3,2).*Ysrc+ BM(3,3).*Zsrc;
end
% function to convert XYZ image to L*a*b* image:
function [L,a,b] = XYZtoLab( X, Y, Z, v )
Xn=v(1,1); % v is the illuminant taken as a row vector.
Yn=v(1,2);
Zn=v(1,3);
% computing L a b from X Y Z and Xn Yn Zn :
if (Y/Yn) > 0.008856
L= 116*((Y/Yn).^(1/3))- 16 ;
else
    L= 903.3*(Y/Yn);
end
a= 500*((X/Xn).^(1/3)- (Y/Yn).^(1/3));
b= 200*((Y/Yn).^(1/3)- (Z/Zn).^(1/3));
end
% function for converting L* a* b* image to XYZ:
function [X,Y,Z] = LabtoXYZ( L, a, b, v )
Xn=v(1,1);% v is the illuminant taken as a row vector.
Yn=v(1,2);
Zn=v(1,3);
if L<= 8.000
    X= Xn*((((L/903.3).^(1/3))+ (a/500)).^3);
    Y= Yn*(L/903.3);
    Z=Zn*((((L/903.3).^(1/3))- (b/200)).^3);
else
    X=Xn*(((L+16)/116)+(a/500)).^(3);
    Y=Yn*(((L+16)/116).^3);
    Z=Zn.*((((L+16)/116)-(b/200)).^3);
end
end
```