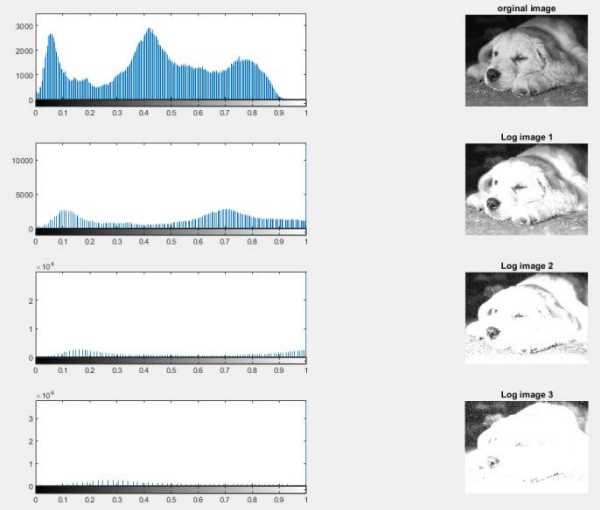
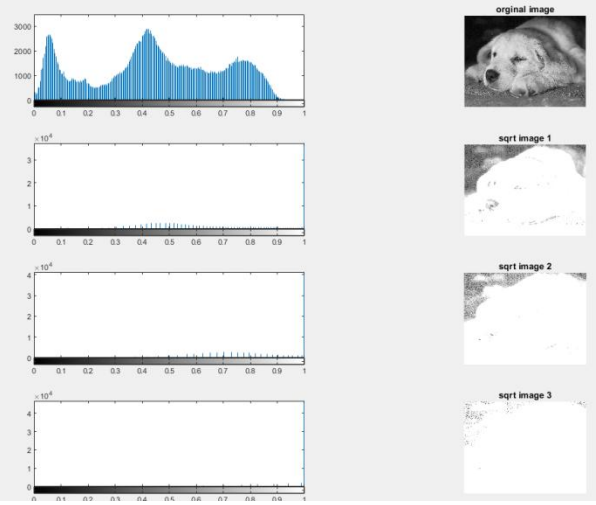
Sprawozdanie Andrzej Żaba nr indeksu 401490

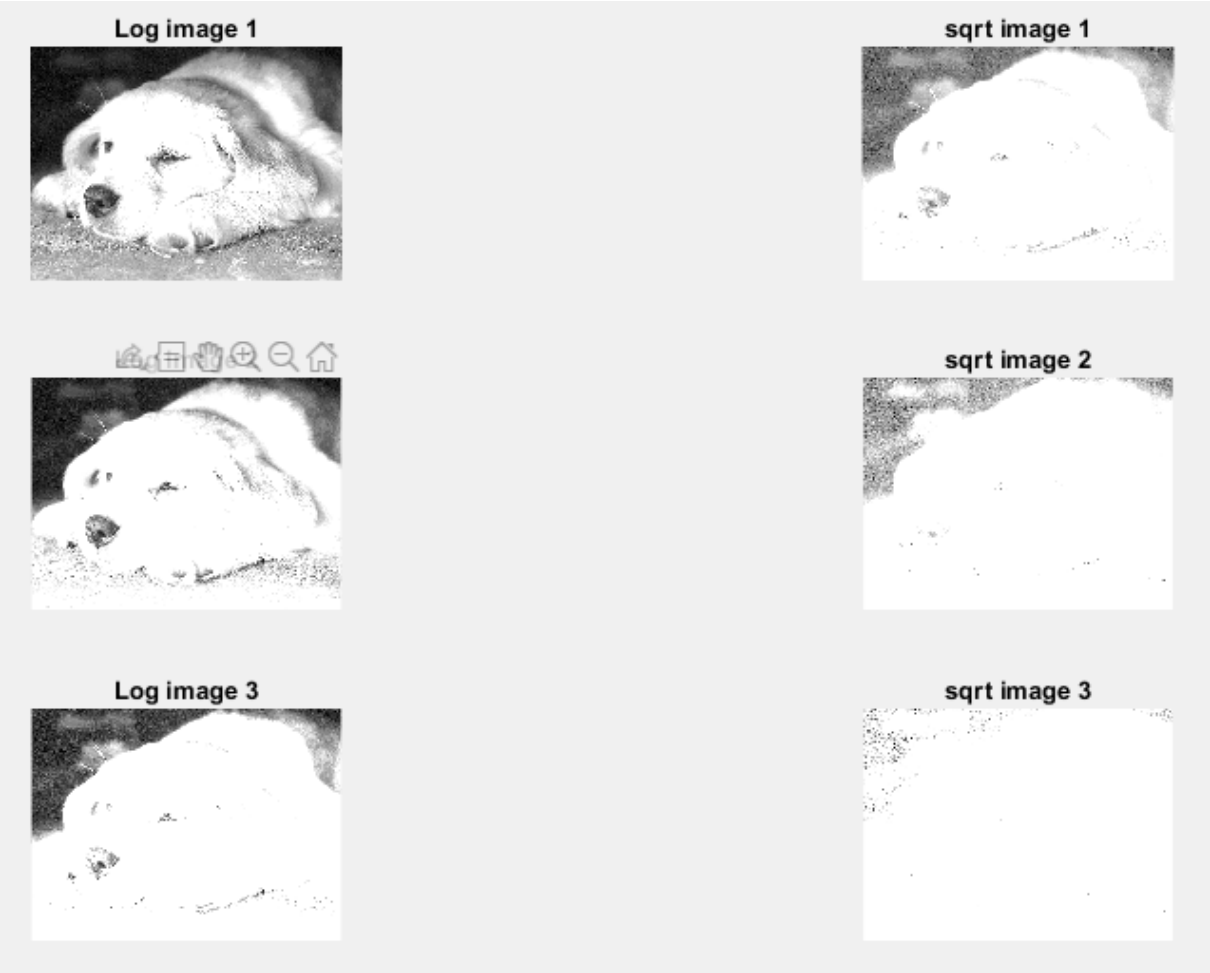
Inżynieria mechatroniczna

Ćwiczenie 1.1

```
y = imfinfo('pies.jpg')
im = imread('pies.jpg');
img = rgb2gray(im);

%imshow(img)

Id = im2double(img);

figure(1)
subplot(4,2,1), imhist(Id);
subplot(4,2,2), imshow(Id); title('orginal image');
%% Log
Out1 = (2*log(1+Id));    % c = 2
Out2 = (3*log(1+Id));    % c = 3
Out3 = (5*log(1+Id));    % c = 5 wraz ze wzrostem c obraz jest bardziej
rozświetlony

subplot(4,2,3), imhist(Out1);
subplot(4,2,4), imshow(Out1); title('Log image 1');

subplot(4,2,5), imhist(Out2);
subplot(4,2,6), imshow(Out2); title('Log image 2');

subplot(4,2,7), imhist(Out3);
subplot(4,2,8), imshow(Out3); title('Log image 3');
%% SQRT

Pierw1 = 2*sqrt(Id);
Pierw2 = 3*sqrt(Id);
Pierw3 = 5*sqrt(Id);

figure(2)

subplot(4,2,1), imhist(Id);
subplot(4,2,2), imshow(Id); title('orginal image');

subplot(4,2,3), imhist(Pierw1);
subplot(4,2,4), imshow(Pierw1); title('sqrt image 1');

subplot(4,2,5), imhist(Pierw2);
subplot(4,2,6), imshow(Pierw2); title('sqrt image 2');

subplot(4,2,7), imhist(Pierw3);
subplot(4,2,8), imshow(Pierw3); title('sqrt image 3');
%% Porównanie
figure(3)

subplot(3,2,1), imshow(Out1); title('Log image 1');
subplot(3,2,3), imshow(Out2); title('Log image 2');
subplot(3,2,5), imshow(Out3); title('Log image 3');

subplot(3,2,2), imshow(Pierw1); title('sqrt image 1');
subplot(3,2,4), imshow(Pierw2); title('sqrt image 2');
subplot(3,2,6), imshow(Pierw3); title('sqrt image 3');
```





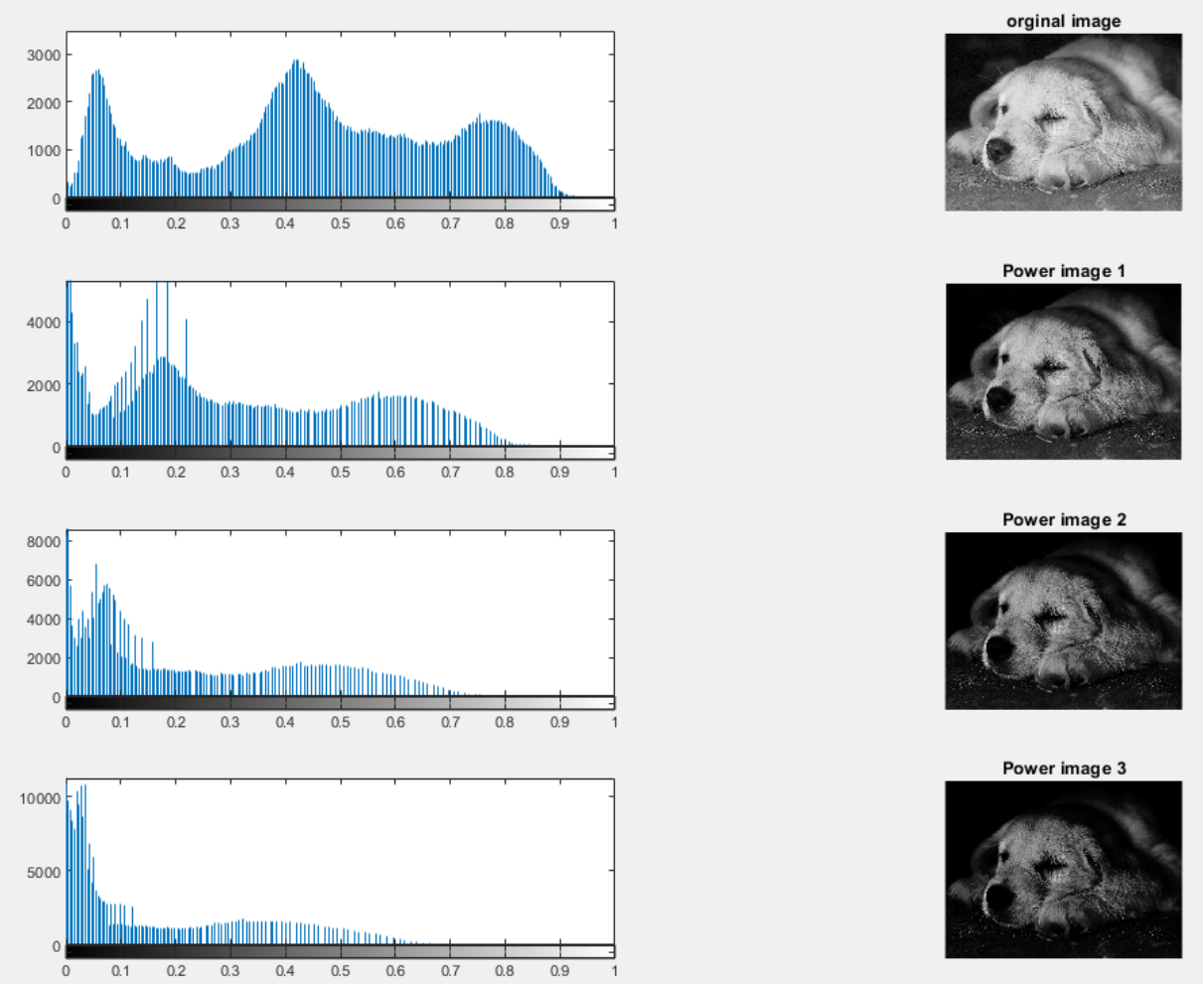


Wraz ze wzrostem stałej **c** obraz jest coraz mocniej rozświetlony.

Ćwiczenie 1.2 , 1.3

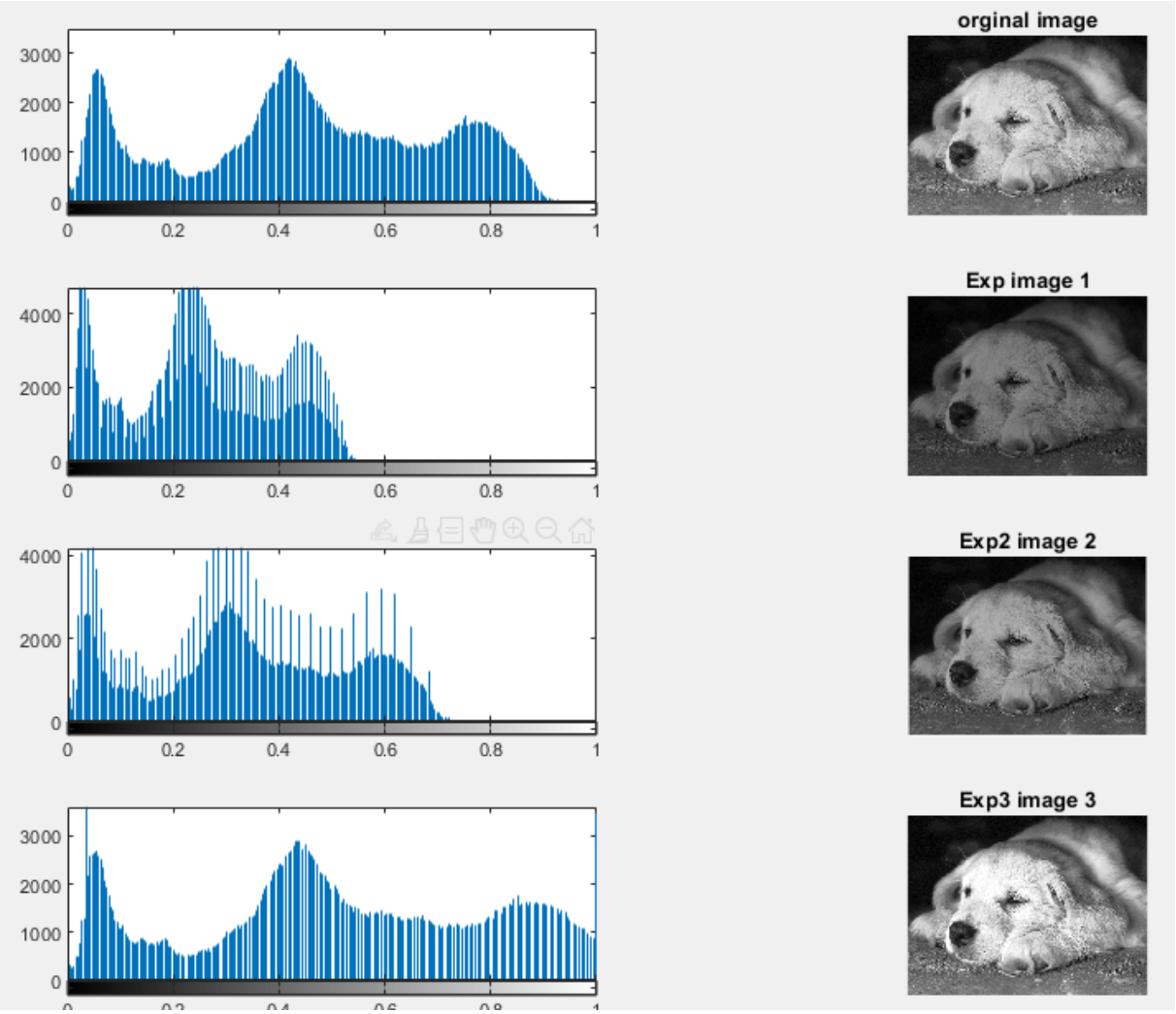
```
clc
clear all
close all

y = imfinfo('pies.jpg');
im = imread('pies.jpg');
img = rgb2gray(im);

Id = im2double(img);

figure(1)
subplot(4,2,1), imhist(Id);
subplot(4,2,2), imshow(Id); title('orginal image');
%% Potega
Out1 = 1*(Id.^2);    % c = 2
Out2 = 1*(Id.^3);    % c = 3
Out3 = 1*(Id.^4);    % c = 4 wraz ze wzrostem c obraz jest bardziej
przyciemniony

subplot(4,2,3), imhist(Out1);
subplot(4,2,4), imshow(Out1); title('Power image 1');

subplot(4,2,5), imhist(Out2);
subplot(4,2,6), imshow(Out2); title('Power image 2');

subplot(4,2,7), imhist(Out3);
subplot(4,2,8), imshow(Out3); title('Power image 3');
%% Exponenta

%ex1 = 0.3*exp(Id);
%ex2 = 0.2*exp(Id);
%ex3 = 0.1*exp(Id);

ex1 = 2*(((1+0.3).^(Id))-1);
ex2 = 2*(((1+0.4).^(Id))-1);
ex3 = 2*(((1+0.6).^(Id))-1);

figure(2)

subplot(4,2,1), imhist(Id);
subplot(4,2,2), imshow(Id); title('orginal image');

subplot(4,2,3), imhist(ex1);
subplot(4,2,4), imshow(ex1); title('Exp image 1');

subplot(4,2,5), imhist(ex2);
subplot(4,2,6), imshow(ex2); title('Exp2 image 2');

subplot(4,2,7), imhist(ex3);
subplot(4,2,8), imshow(ex3); title('Exp3 image 3');
%% Porównanie
figure(3)
subplot(3,2,1), imshow(Out1); title('Power image 1');
subplot(3,2,3), imshow(Out2); title('Power image 2');
subplot(3,2,5), imshow(Out3); title('Power image 3');

subplot(3,2,2), imshow(ex1); title('Exp image 1');
subplot(3,2,4), imshow(ex2); title('Exp image 2');
subplot(3,2,6), imshow(ex3); title('Exp image 3');
```

Histogramy i wykresy dla funkcji potęgowej.



Wraz ze wzrostem potęgi do której wyrażenie jest podnoszone przyciemnienie obrazu jest mocniejsze. Z każdym kolejnym obrazkiem widać stopniowe przesunięcia w lewo wartości na histogramie – czyli w stronę ciemniejszych odcieni.

Histogramy i obrazy dla funkcji ekspotencjalnej:

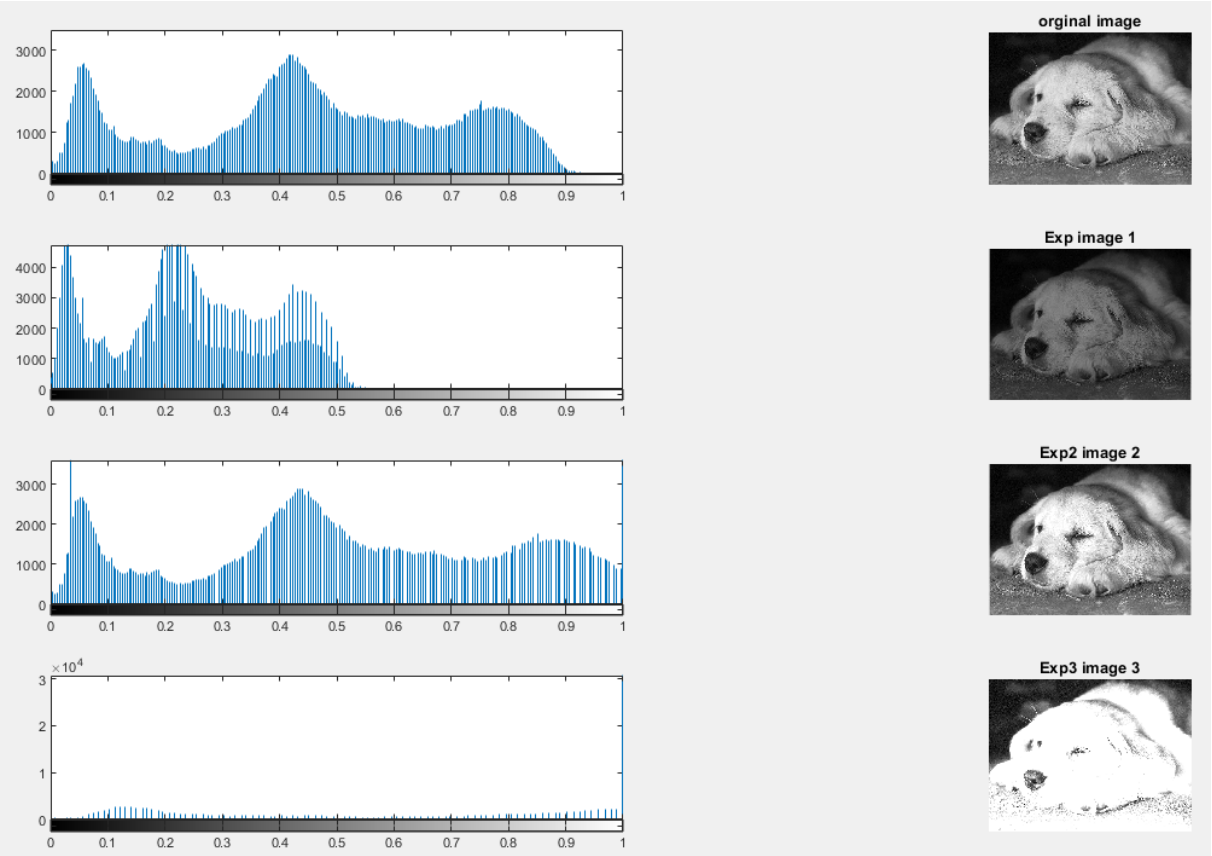


Dla niskich wartości alfa (np. 0,1) otrzymujemy przyciemnienie obrazu. Natomiast stopniowo ze zwiększaniem tego parametru obraz staje się jaśniejszy. Już przy wartości alfa = 0,6 widzimy wyraźne rozjaśnienie w stosunku do oryginalnego obrazu – również dobrze widoczne na histogramach.

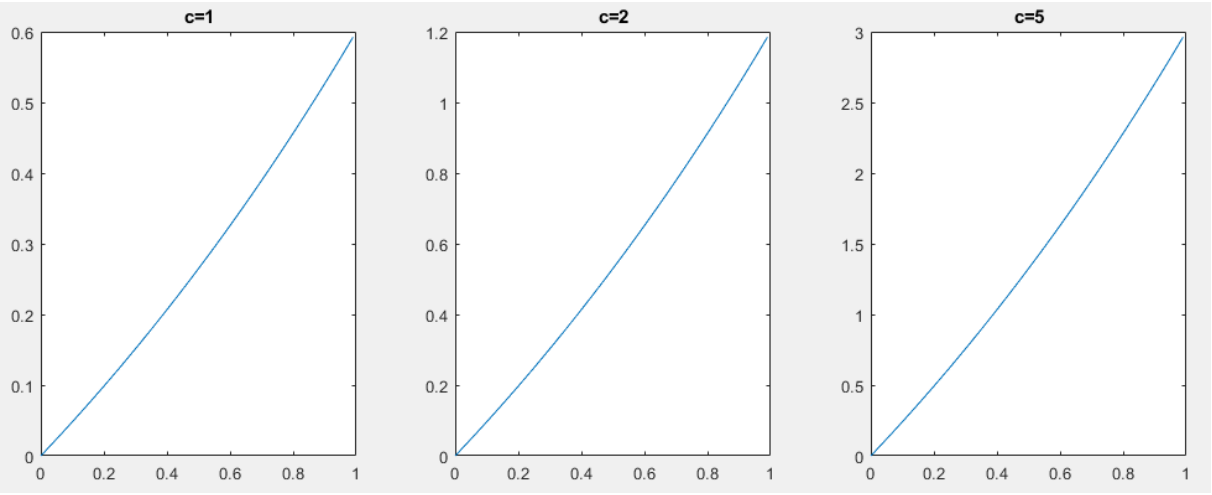
Sprawdzono również wpływ stałej c dla podanych wartości:

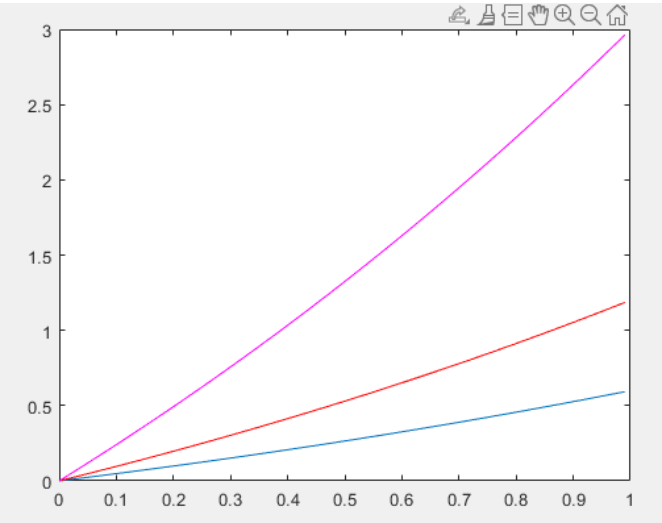
```
ex1 = 1*(((1+0.6).^(Id))-1);
ex2 = 2*(((1+0.6).^(Id))-1);
ex3 = 5*(((1+0.6).^(Id))-1);
```

Histogramy i obrazy dla funkcji ekspotencjalnej (badanie parametru c)



Postacie funkcji eksponencjalnej dla takiej samej wartości **alfa** lecz dla różnych wartościach parametru **c**.



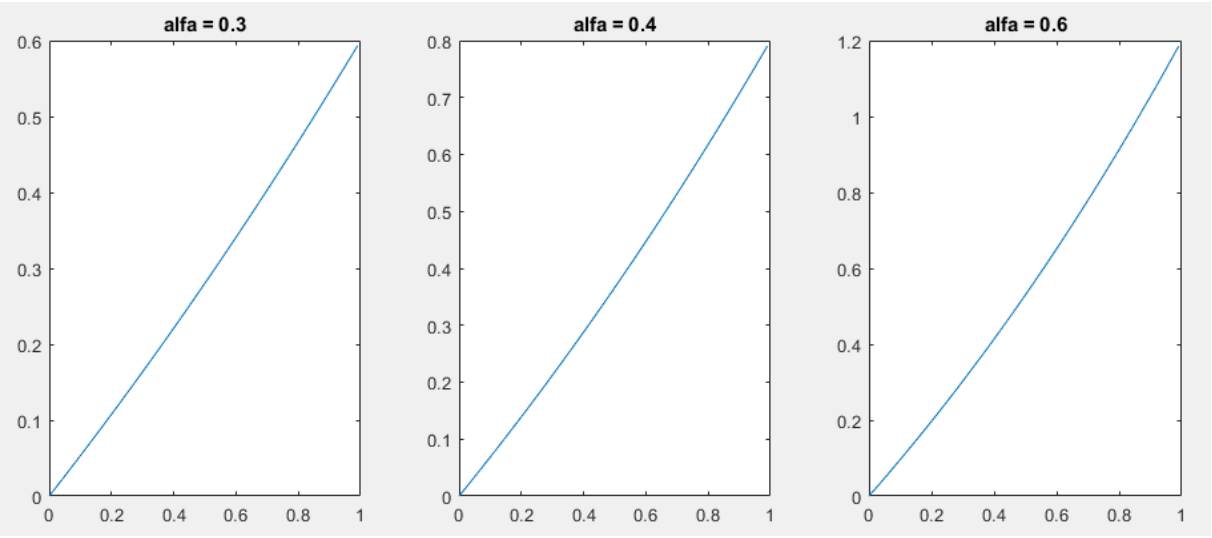


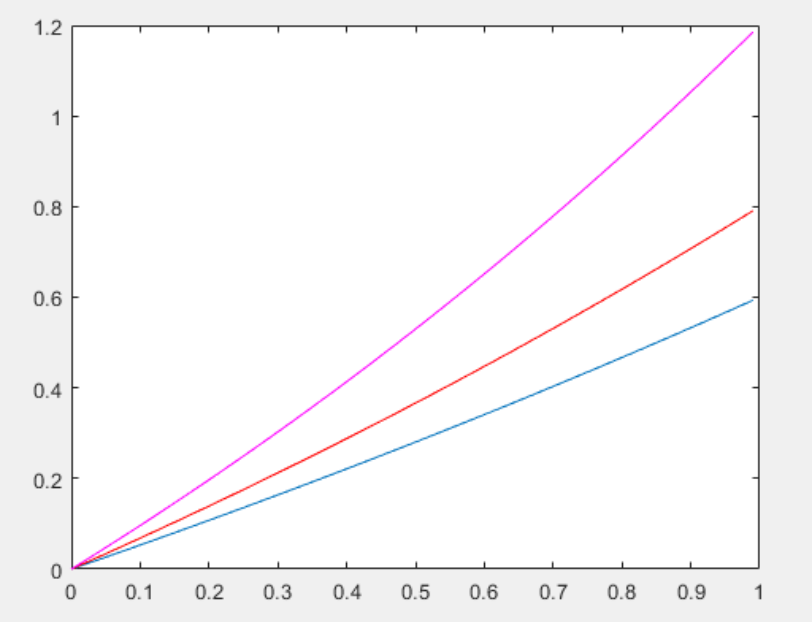
Różowy – c =5

czerwony – c = 2

niebieski – c= 1

Postacie funkcji eksponencjalnej dla takiej samej wartości **c** lecz dla różnych wartościach parametru **alfa**.



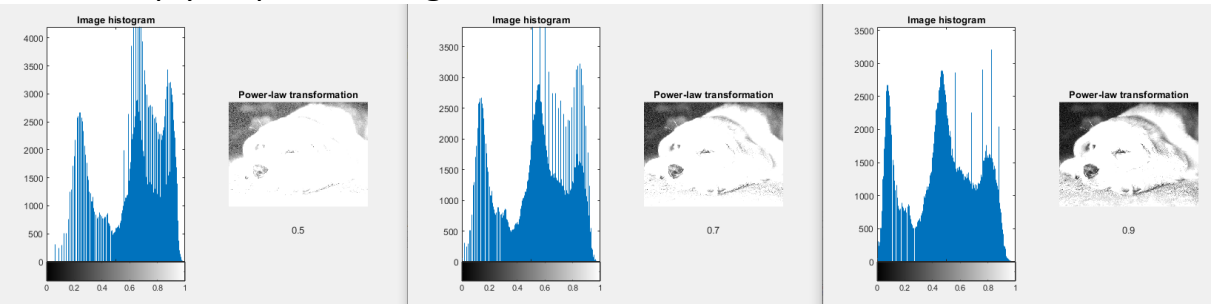


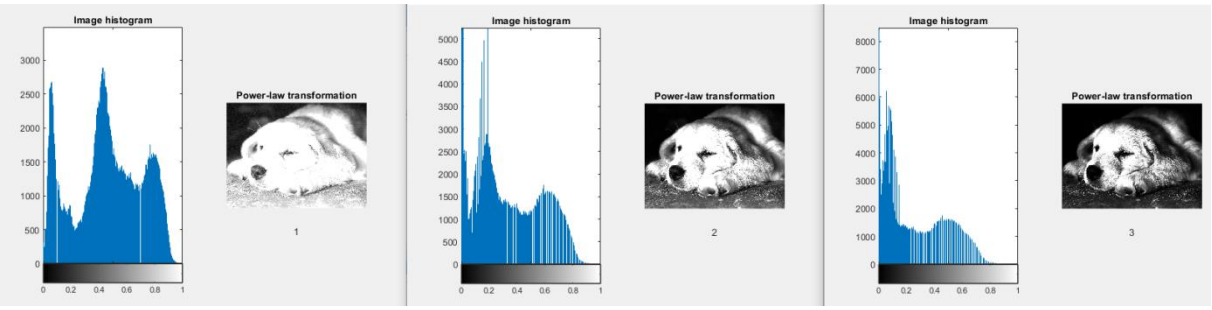
Różowy – alfa = 0.6
czerwony – alfa = 0.4
niebieski – alfa = 0.3

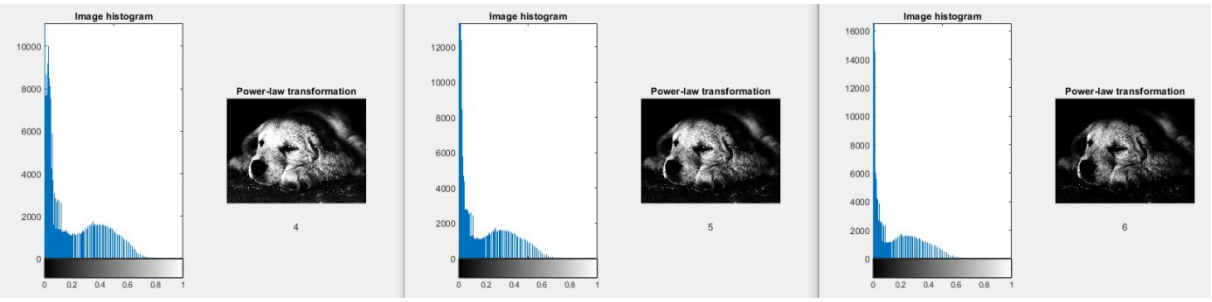
Zad 1.4

```
clc
clear all
close all

y = imfinfo('pies.jpg');
im = imread('pies.jpg');
img = rgb2gray(im);

Id = im2double(img);
[m,n] = size(Id);

%Rownanie s = c*(r^gamma)

c = 4; % im większe C tym obraz na koncu jasniejszy
g = [0.5 0.7 0.9 1 2 3 4 5 6]; %Gamma Array

    % dla g<1 - rozjasnienie dla g>1 - przyciemnienie

for r=1:length(g)
    for p=1:m
        for q=1:n
            Im2(p,q) = c*Id(p,q).^g(r);
        end
    end

figure(r)
subplot(1,2,1),imhist(mat2gray(Im2)); title('Image
histogram'); xlabel('Gamma='),xlabel(g(r));
subplot(1,2,2),imshow(Im2); title('Power-law
transformation'); xlabel('Gamma='),xlabel(g(r));

end
```

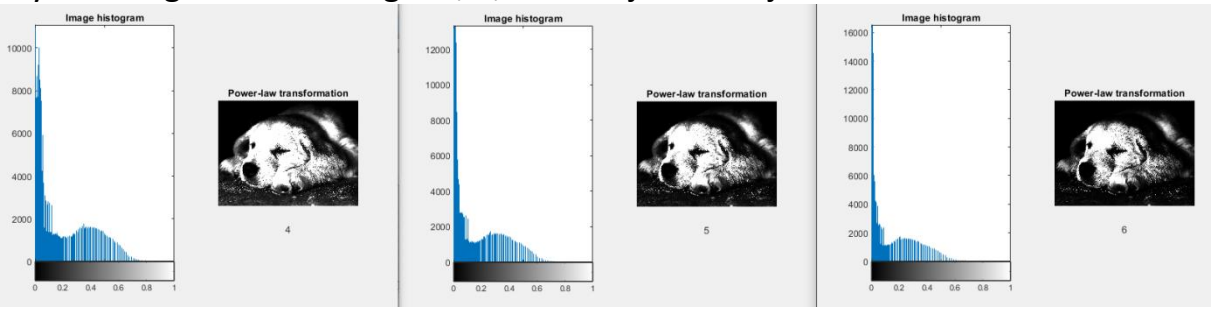
Badanie wpływu parametru **gamma**:







Wyniki dla gamma równego: 4, 5, 6 i stałej **c** równej 4



Wraz ze wzrostem parametru **c** następuje rozjaśnienie obrazu. Natomiast jeżeli parametr **gamma** jest mniejszy od 1 powoduje on rozjaśnienie obrazu, jeżeli jednak jest on większy od 1 – przyciemnia obraz.

Zad 1.5.1

```
clc
clear all
close all

y = imfinfo('pies.jpg');
im = imread('pies.jpg');
img = rgb2gray(im);

img = im2double(img);
figure(1)
subplot(1,2,1), imshow(img); title('Input image'); %
Display Image

Im2 = imadjust(img,[0 1],[0 1],1./3);
% Map input gray values of image A in
% range 0-1 to an output range of 0-1
% with gamma factor of 1/3 (i.e r =3)

subplot(1,2,2), imshow(Im2), title('Output image');
%results
```

Tak prezentuje się obraz po zmianie skali szarości zaimplementowanej w kodzie powyżej:

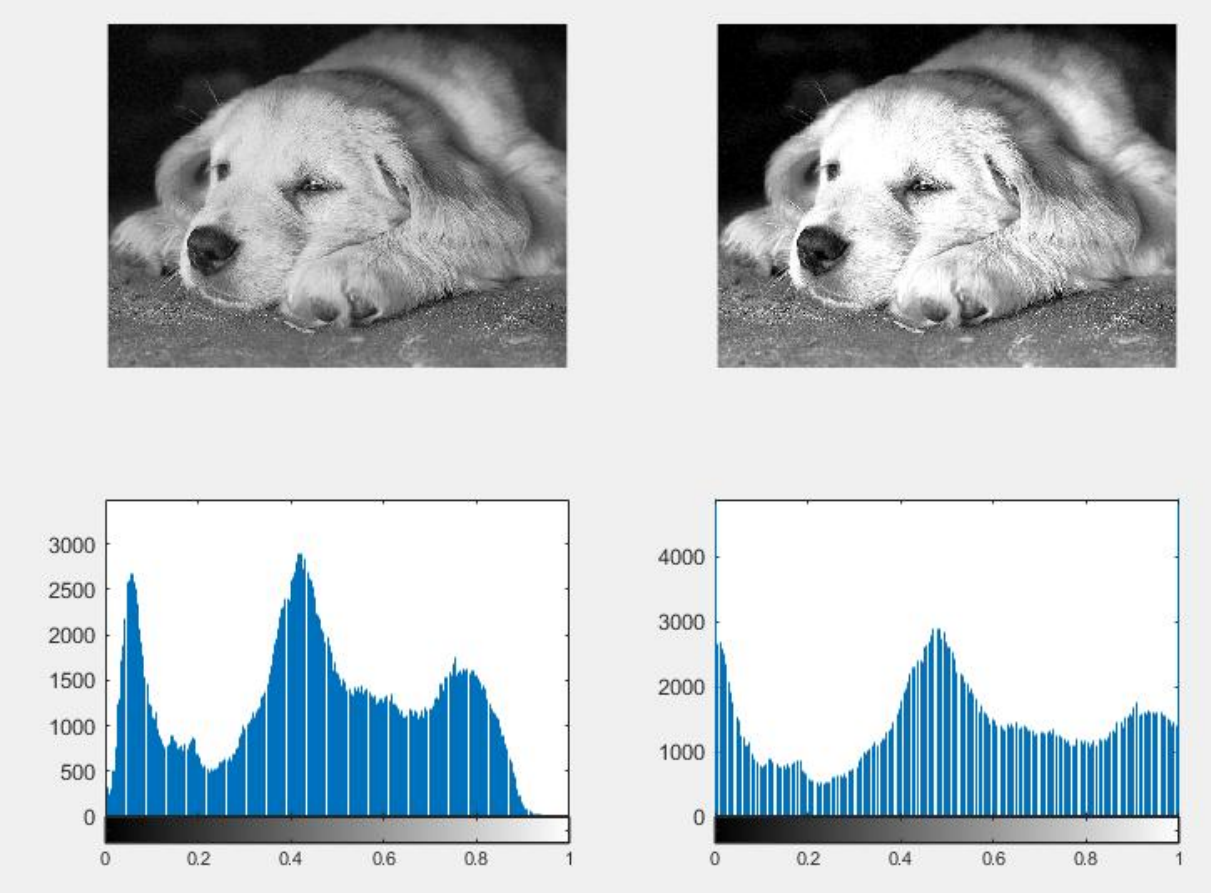
Zad 1.5.2

```
ImInv = imadjust(img,[0 1],[1 0]);
figure(2)
subplot(1,2,1), imshow(img); title('Input image'); %
Display Image
subplot(1,2,2), imshow(ImInv), title('Output image');
%results
```

Poprawnie uzyskano inwersję obrazu za pomocą komendy **imadjust**.

Zad 1.5.3

```
clc
clear all
close all

y = imfinfo('pies.jpg');
im = imread('pies.jpg');
img = rgb2gray(im);

img = im2double(img);

Is = imadjust(img, stretchlim(img, [0.05 0.95]),[]);

figure(1)
subplot(2,2,1), imshow(img);

subplot(2,2,2), imshow(Is);

subplot(2,2,3), imhist(img);

subplot(2,2,4), imhist(Is);
```



Używając powyżej zaimplementowanej metody udało się poprawnie rozciągnąć zakres poziomów jasności. Widać to dobrze na histogramie po prawej stronie, gdzie wartości sięgają już po samą ‘1’. Przejawia się to na obrazku występowaniem czysto białych pikseli (w okolicach głowy pieska).

Zad 1.5.4

```
Is4 = imadjust(img, [],[],.2);
Is5 = imadjust(img, [],[],2);
figure(2)
subplot(1,2,1)
imshow(Is4);
title('gamma = 0.2')
subplot(1,2,2)
imshow(Is5);
title('gamma = 2')
```

Po dodaniu powyższego kodu otrzymujemy następujące wyniki – potwierdzające, że **gamma** < 1 – rozjaśnia a **gamma** > 1 – przyciemnia obraz.

Zad 1.5.5

```
Is6 = imadjust(img, [],[1 0],.2);
Is7 = imadjust(img, [],[1 0],2);
figure(3)
subplot(1,2,1)
imshow(Is6);
title('gamma = 0.2')
subplot(1,2,2)
imshow(Is7);
title('gamma = 2')
```

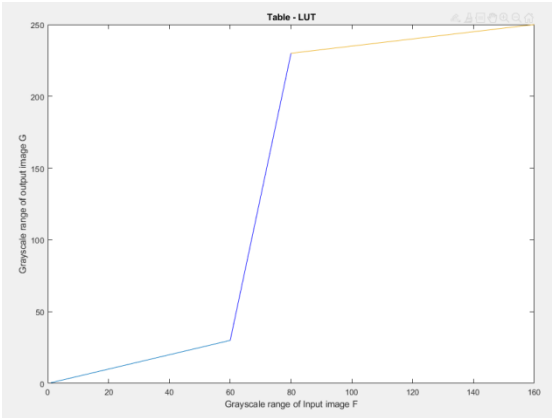
Gdy zastosujemy inwersję obrazu parametr gamma również będzie działał odwrotnie niż poprzednio. Gdy **gamma** < 1 – przyciemnia a gdy **gamma** > 1 – rozjaśnia obraz.

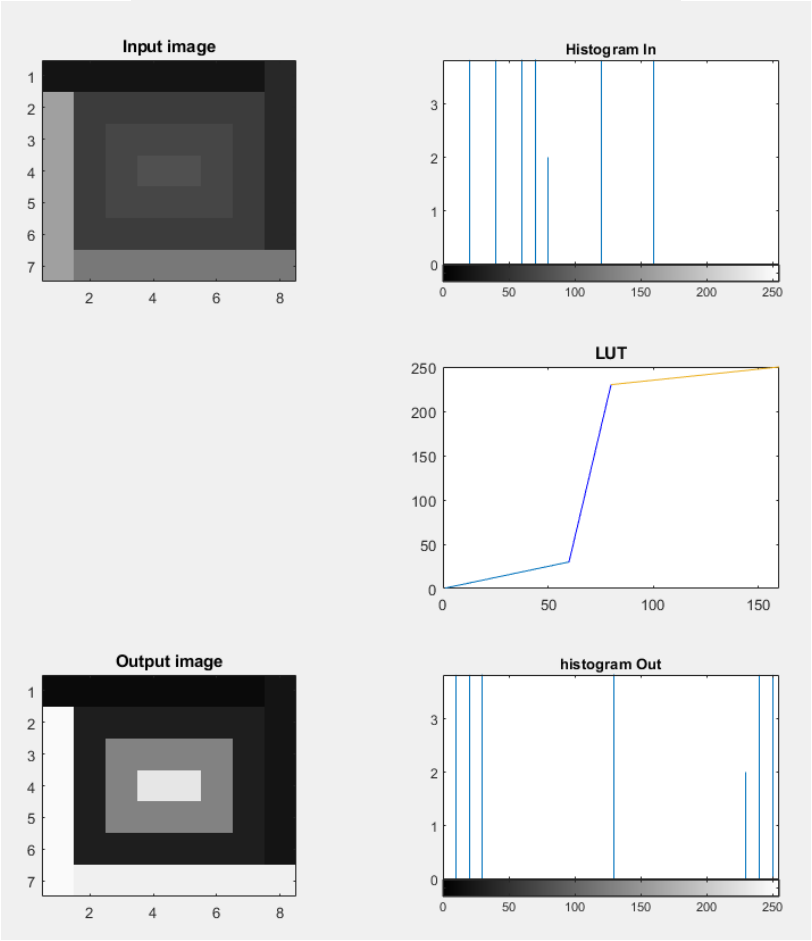
Zad 1.6.1

```
clc
clear all
close all

lut = [];
x = [];      %input image
y = [];      %output image

x = [20 20 20 20 20 20 20 40;
    160 60 60 60 60 60 60 40;
    160 60 70 70 70 70 60 40;
    160 60 70 80 80 70 60 40;
    160 60 70 70 70 70 60 40;
    160 60 60 60 60 60 60 40;
    160 120 120 120 120 120 120 120];

[m,n] = size(x);

for i=1:60
    lut(i) = 0.5*i;
end

for i = 60:80
    lut(i) = 10*(i-57);
end

for i= 80:160
    lut(i) = 0.25*i + 210;
end

%Gary level modification using LUT
for i = 1:m
    for j = 1:n

        y(i,j)=lut(x(i,j));
    end
end

%display LUT
t = 1:60;
t1 = 60:80;
t2 = 80:160;

plot(t,lut(t),t1,lut(t1),'b',t2,lut(t2)),title('Table - LUT')
xlabel('Grayscale range of Input image F')
ylabel('Grayscale range of output image G')

%Display input image, histogram and LUT
figure(2)
subplot(331), subimage(uint8(x),gray(256)),title('Input image')
subplot(332), imhist(uint8(x)),title('Histogram In')
subplot(335), plot(t,lut(t),t1,lut(t1),'b-',t2,lut(t2)),
title("LUT")
```

```
%display output image, histogram
subplot(337), subimage(uint8(y),gray(256)),title('Output image')
subplot(338), imhist(uint8(y)), title('histogram Out')
```

Powyższym kodem zrealizowano przykład przedstawiony w instrukcji do laboratoriów.

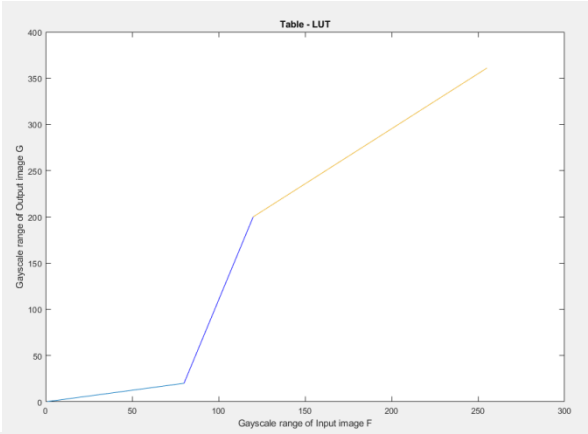


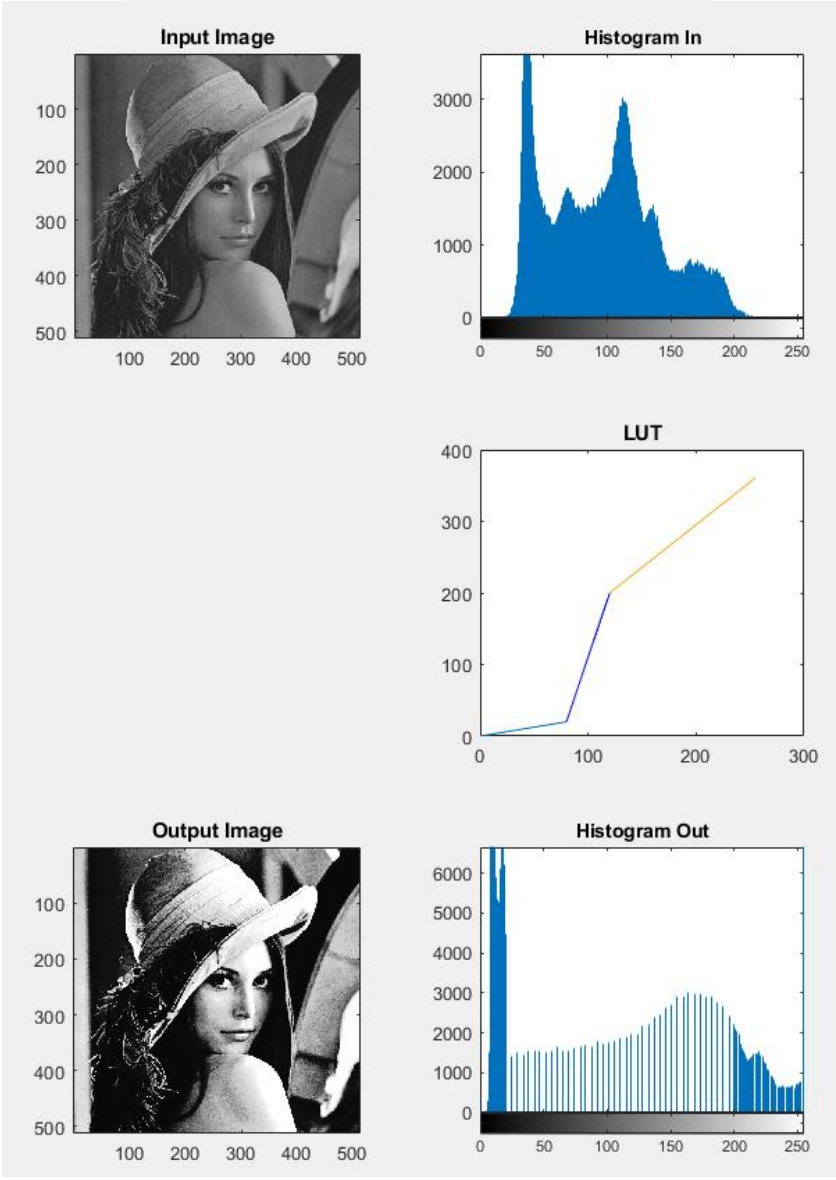


Zad 1.6.2

```
clc
clear all
close all

op = imfinfo('Lena.jpg');
[x, map] = imread('Lena.jpg');

x = rgb2gray(x);

%x = im2double(img);
%comp     :0-80 -> 0-20
%strech     :80-120 -> 20-200
%strech     :120-227 -> 200-256

%to find max value of Image
% max(max(Image_in))

%to find image resolution:
[m, n] = size(x);

for i=1:1:80
    lut(i) = (20/80)*i;                          %comp:  0-80 ->0:20
end

for i = 80:1:120
    lut(i) = (180/40)*(i-80) + 20;               % strech: 80-120 ->
20-200
end

for i = 120:1:255
    lut(i) = (56/47)*(i-120) + 200;              % strech: 120-227 ->
256
end

% gray level modification using LUT
for i=1:m
    for j=1:n
        %gray values of input image are index for LUT table
        % Image_out(y) = LUT(Image_in)
        y(i,j) = lut(x(i,j));
    end
end

t=1:80;
t1 = 80:120;
t2 = 120:255;

plot(t,lut(t),t1,lut(t1),'b-',t2,lut(t2)),title('Table - LUT')
xlabel('Gayscale range of Input image F')
ylabel('Gayscale range of Output image G')

figure(2)
% I
subplot(331),subimage(uint8(x),gray(256)),title('Input Image')
```

```
subplot(332),imhist(uint8(x),gray(256)),title('Histogram In')
subplot(335),plot(t,lut(t),t1,lut(t1),'b-',t2,lut(t2)),title('LUT')
%y

subplot(337),subimage(uint8(y),gray(256)),title('Output Image')
subplot(338),imhist(uint8(y),gray(256)),title('Histogram Out')
```

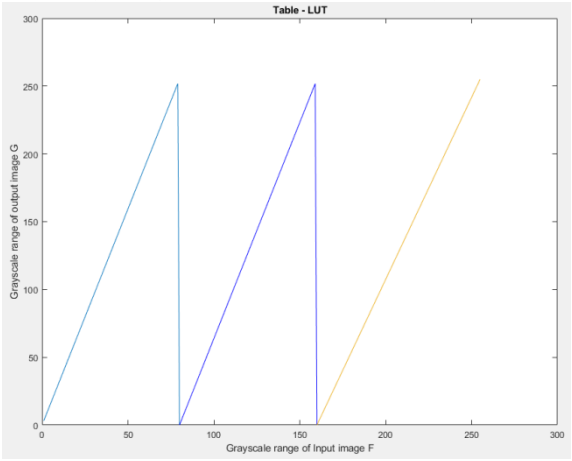




Zad 1.6.3

```
clc
clear all
close all

lut = [];
x = [];      %input image
y = [];      %output image


x = [20 20 20 20 20 20 20 80;
     160 60 60 60 60 60 60 40;
     160 60 70 70 70 70 60 40;
     160 60 70 80 80 70 60 80;
     160 60 70 70 70 70 60 40;
     160 60 60 60 60 60 60 40;
     120 100 80 60 40 20 10 80;
     160 120 120 120 120 120 120 120];

[m,n] = size(x);
```

Rozpoczęto od wyczyszczenia Workspace'a, oraz inicjacji tablic **x**,**y**,**lut**. Następnie stworzono obraz syntetyczny i zapisano go jako **x**. Wartościom **m** i **n** przypisano wartości wysokości i szerokości obrazu.



Chcący uzyskać funkcję LUT jak na powyższym obrazie zaimplementowano poniższy kod:

```
for i = 1:80
    lut(i) = 255/80*i;
end

for i = 80:160
    lut(i) = 255/80*i - 255;
end

for i = 160:255
```

```
    lut(i) = 255/95*i - 429.473;
end
```

Skorzystano z poniższego wzoru odejmując odpowiednie wartości aby uzyskać odpowiedni wykres:

F(I,j) = [Rmax – Rmin /Gmax – Gmin]* (G(I,j)-Gmin) + Rmin

where: Gmax/min - max/min wartości poz. szarości w obrazie źródłowym

Rmax/min - max/min wartości możliwych poz. szarości

Następnie zaimplementowano poniższy kod.

```
% Gary level modification using LUT
for i = 1:m
    for j = 1:n
        y(i,j) = lut(x(i,j));
    end
end
```

Odpowiada on za stworzenie nowego obrazu. Kod ten dla każdego pikselu w obrazie x pobiera jego wartość (**x(i,j)**) a następnie w zależności od jego wartości używa go jako argumentu **i** dla funkcji **lut().** Wtedy do odpowiedniego miejsca w tablicy **y** zostaje przypisana odpowiednia wartość skali szarości.

Następnie zaimplementowano poniższy kod w celu pokazania postaci funkcji lut – według której zmieniony został nasz obraz.

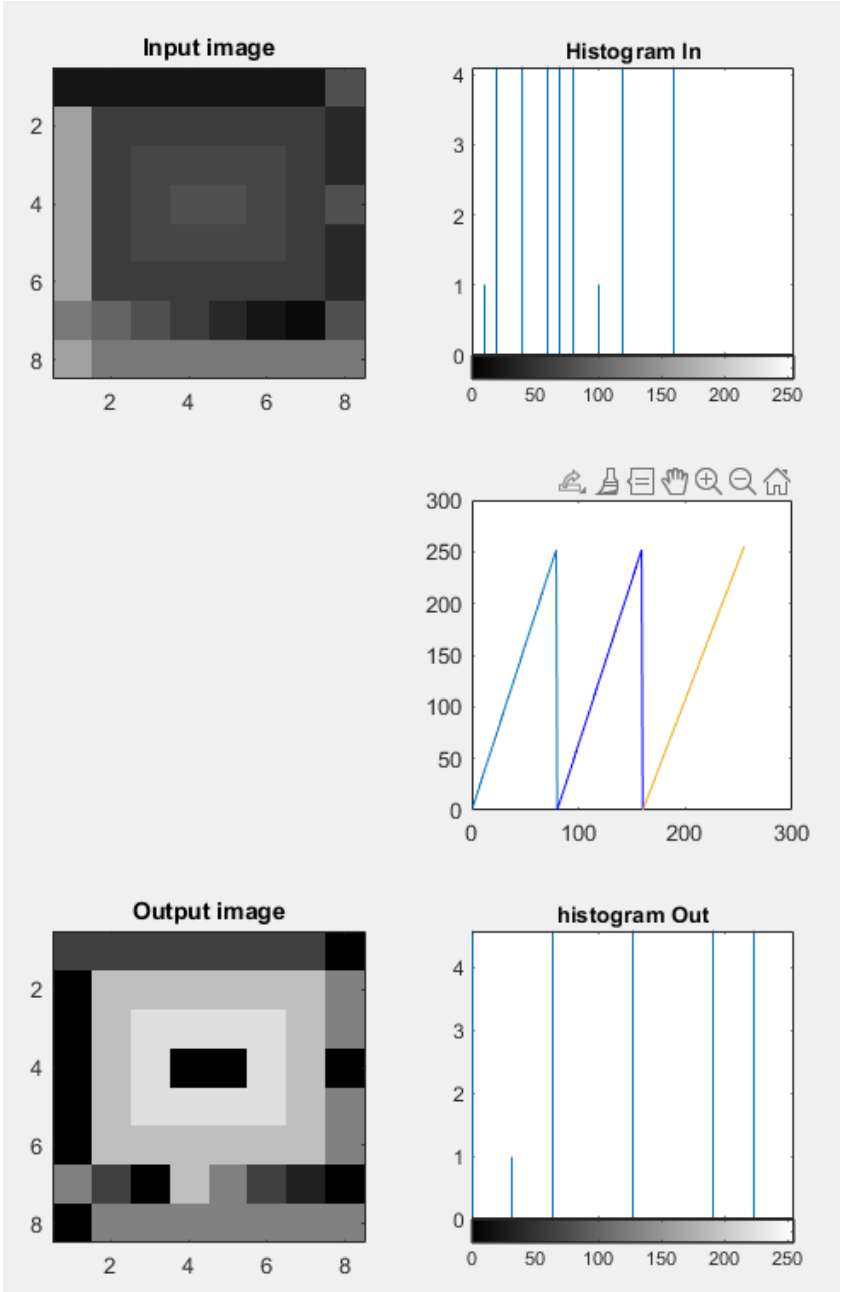
```
%Display LUT
t = 1:80;
t1 = 80:160;
t2 = 160:255;

plot(t,lut(t),t1,lut(t1),'b',t2,lut(t2)),title('Table
- LUT')
xlabel('Grayscale range of Input image F')
ylabel('Grayscale range of output image G')
```

Na końcu dzięki poniższym linijkom zostały pokazane: obraz oryginalny, obraz po modyfikacji skali szarości, ich histogramy oraz wykres funkcji lut.

```
%Display input image, histogram and LUT
figure(2)
subplot(331),
subimage(uint8(x),gray(256)),title('Input image')
```

```
subplot(332), imhist(uint8(x)),title('Histogram In')
subplot(335), plot(t,lut(t),t1,lut(t1),'b-
',t2,lut(t2)), title("LUT")

%display output image, histogram
subplot(337),
subimage(uint8(y),gray(256)),title('Output image')
subplot(338), imhist(uint8(y)), title('histogram
Out')
```

Otrzymano wynik:



Całość kodu:

```
clc
clear all
close all

lut = [];
x = [];      %input image
y = [];      %output image

x = [20 20 20 20 20 20 20 80;
    160 60 60 60 60 60 60 40;
    160 60 70 70 70 70 60 40;
    160 60 70 80 80 70 60 80;
    160 60 70 70 70 70 60 40;
    160 60 60 60 60 60 60 40;
    120 100 80 60 40 20 10 80;
    160 120 120 120 120 120 120 120];

[m,n] = size(x);

for i = 1:80
    lut(i) = 255/80*i;
end

for i = 80:160
    lut(i) = 255/80*i - 255;
end

for i = 160:255
    lut(i) = 255/95*i - 429.473;
end

% Gary level modification using LUT
for i = 1:m
    for j = 1:n
        y(i,j) = lut(x(i,j));
    end
end

%Display LUT
t = 1:80;
t1 = 80:160;
t2 = 160:255;

plot(t,lut(t),t1,lut(t1),'b',t2,lut(t2)),title('Table - LUT')
xlabel('Grayscale range of Input image F')
ylabel('Grayscale range of output image G')

%Display input image, histogram and LUT
figure(2)
subplot(331), subimage(uint8(x),gray(256)),title('Input image')
subplot(332), imhist(uint8(x)),title('Histogram In')
subplot(335), plot(t,lut(t),t1,lut(t1),'b-',t2,lut(t2)), title("LUT")

%display output image, histogram
subplot(337), subimage(uint8(y),gray(256)),title('Output image')
subplot(338), imhist(uint8(y)), title('histogram Out')
```

Zad 1.6.4

W zadaniu zastosowano taką samą metodę co w zad 1.6.3 jednak wykorzystano do tego obraz rzeczywisty.

Analogicznie do poprzedniego zadania do tablicy **x** przypisano obraz (tutaj czarno biały w którym piksele przyjmują wartości od 0 do 250). Ponownie do zmiennych **n** i **m** przypisano wymiary obrazu. W komentarzu wypisano wartości skali szarości z których chcemy uzyskać kolejne.

```
clc
clear all
close all

op = imfinfo('pies.jpg');
[x, map] = imread('pies.jpg');

x = rgb2gray(x);

%x = im2double(img);
%strech      :0-80 -> 0-255
%strech      :80-160 -> 0-255
%strech      :160-255 -> 0-255

%to find max value of Image
% max(max(Image_in))

%to find image resolution:
[m, n] = size(x);
```

Kolejno, użyto tych samych pętli do przypisania funkcji do zmiennej lut w celu stworzenia odpowiedniej tablicy na podstawie której w następnym etapie stworzono obraz przypisany do **y**.

```
for i = 1:80
    lut(i) = 255/80*i;
end

for i = 80:160
    lut(i) = 255/80*i - 255;
end

for i = 160:255
    lut(i) = 255/95*i - 429.473;
end
```

```
% gray level modification using LUT
for i=1:m
    for j=1:n
        %gray values of input image are index for LUT
table
        % Image_out(y) = LUT(Image_in)
        y(i,j) = lut(x(i,j)+1);
    end
end

t = 1:80;
t1 = 80:160;
t2 = 160:255;
```

Na koniec ponownie wykreślono na wykresach postać funkcji **lut()** jak również: obraz przed i po modyfikacji skali szarości oraz ich histogramy:

```
plot(t,lut(t),t1,lut(t1),'b-
',t2,lut(t2)),title('Table - LUT')
xlabel('Gayscale range of Input image F')
ylabel('Gayscale range of Output image G')

figure(2)
% I
subplot(331),subimage(uint8(x),gray(256)),title('Inpu
t Image')
subplot(332),imhist(uint8(x),gray(256)),title('Histog
ram In')
subplot(335),plot(t,lut(t),t1,lut(t1),'b-
',t2,lut(t2)),title('LUT')
%y
subplot(337),subimage(uint8(y),gray(256)),title('Outp
ut Image')
subplot(338),imhist(uint8(y),gray(256)),title('Histog
ram Out')
```

Wyniki programu prezentują się następująco:

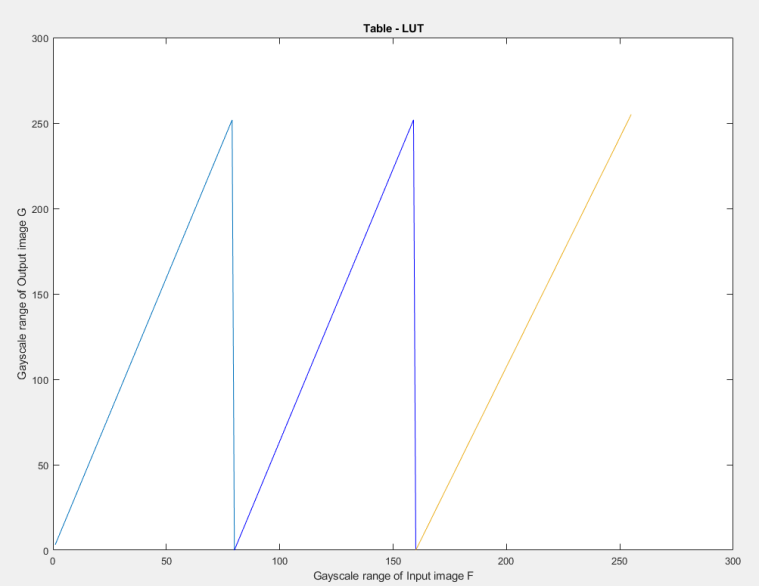
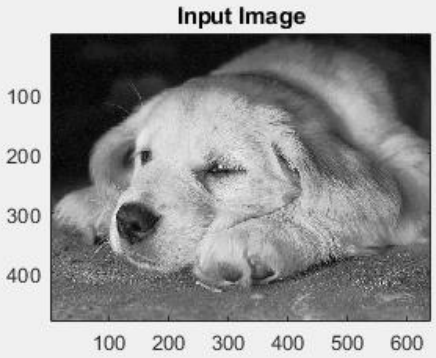
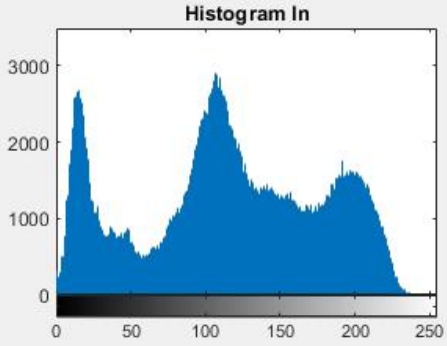
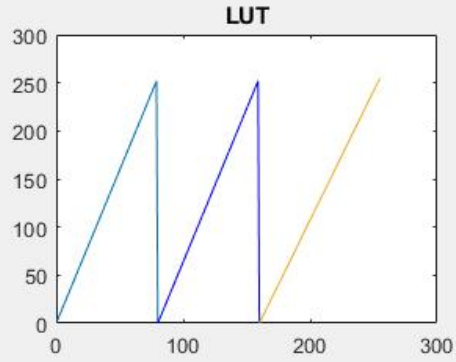
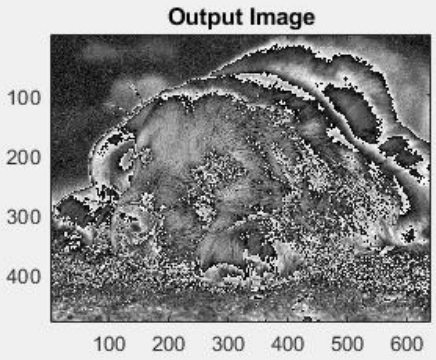


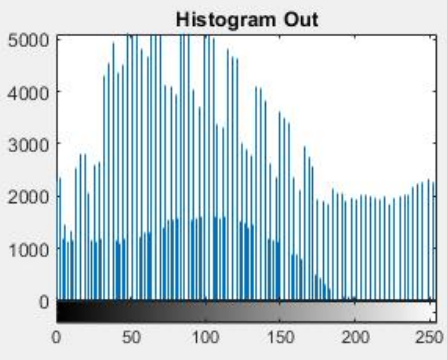
Table - LUT
Gayscale range of Output image G
Gayscale range of Input image F
0
50
100
150
200
250
300


Input Image
100
200
300
400
100 200 300 400 500 600


Histogram In
3000
2000
1000
0
0 50 100 150 200 250


LUT
300
250
200
150
100
50
0
0 100 200 300


Output Image
100
200
300
400
100 200 300 400 500 600


Histogram Out
5000
4000
3000
2000
1000
0
0 50 100 150 200 250