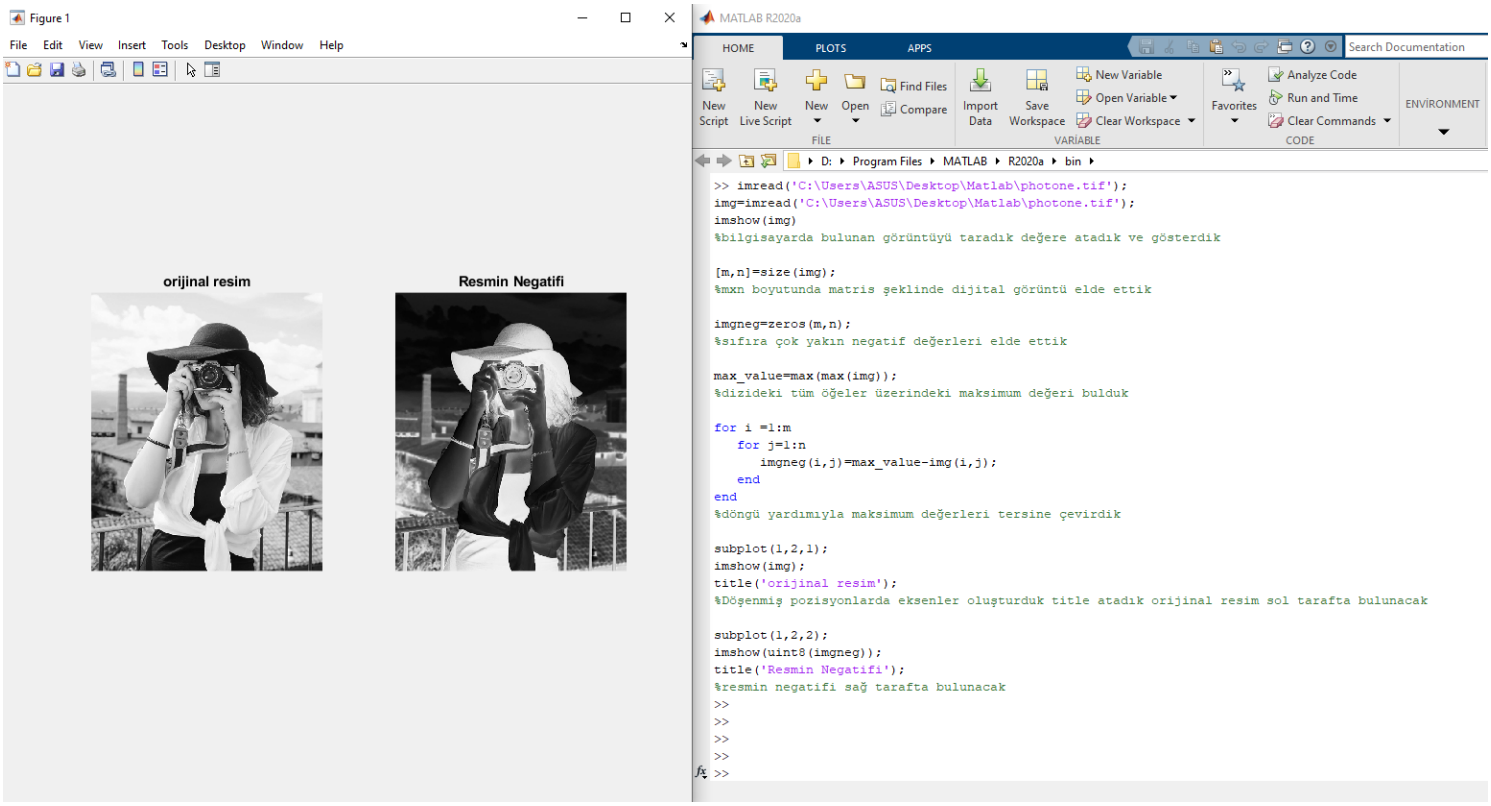
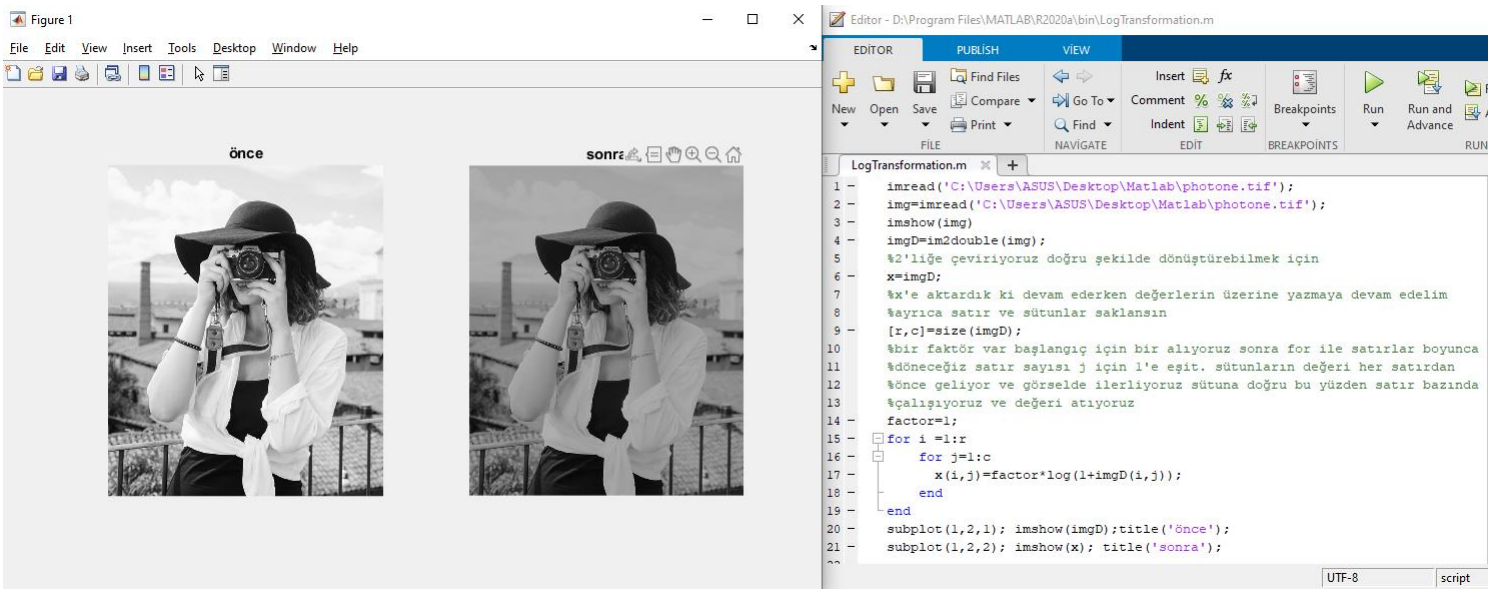
----------Negative of Image----------




```
imread('C:\Users\ASUS\Desktop\Matlab\photone.tif');
img=imread('C:\Users\ASUS\Desktop\Matlab\photone.tif');
imshow(img)
%bilgisayarda bulunan görüntüyü taradık değere atadık ve gösterdik

[m,n]=size(img);
%mxn boyutunda matris şeklinde dijital görüntü elde ettik

imgneg=zeros(m,n);
%sıfıra çok yakın negatif değerleri elde ettik

max_value=max(max(img));
%dizideki tüm öğeler üzerindeki maksimum değeri bulduk

for i =1:m
  for j=1:n
    imgneg(i,j)=max_value-img(i,j);
  end
end
%döngü yardımıyla maksimum değerleri tersine çevirdik

subplot(1,2,1);
imshow(img);
title('orijinal resim');
%Döşenmiş pozisyonlarda eksenler oluşturduk title atadık orijinal resim sol tarafta bulunacak
```

```
subplot(1,2,2);
imshow(uint8(imgneg));
title('Resmin Negatifi');
%resmin negatifi sağ tarafta bulunacak
```

----------Log Transformation----------

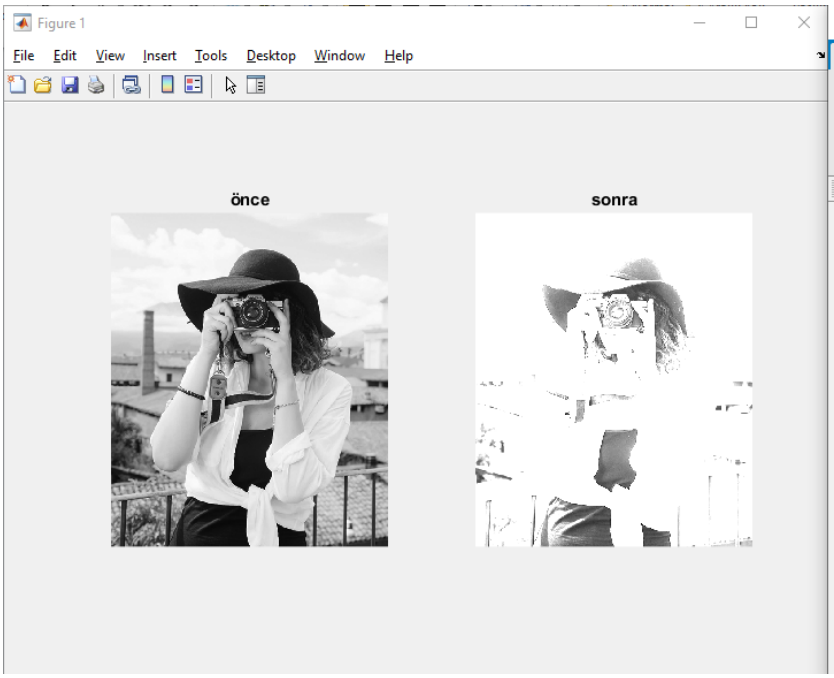


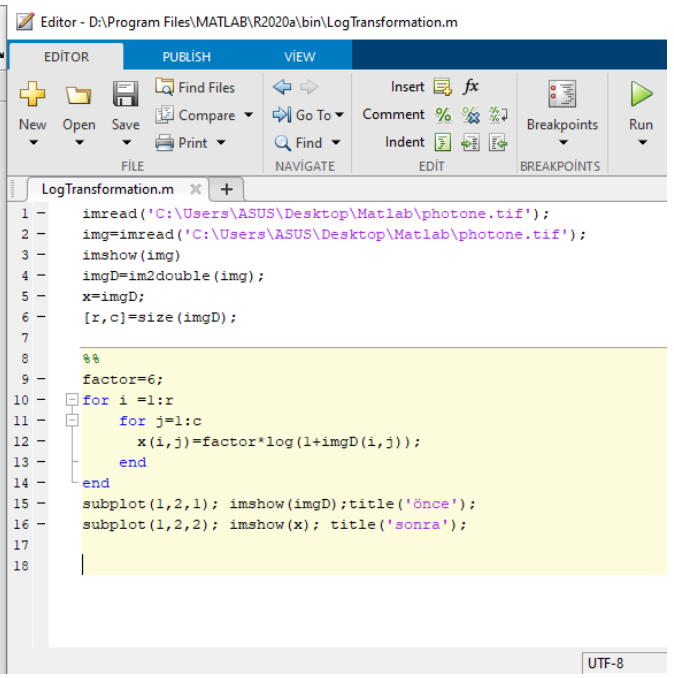

```
imread('C:\Users\ASUS\Desktop\Matlab\photone.tif');
img=imread('C:\Users\ASUS\Desktop\Matlab\photone.tif');
imshow(img)
imgD=im2double(img);
%2'liğe çeviriyoruz doğru şekilde dönüştürebilmek için

x=imgD;
%x'e aktardık ki devam ederken değerlerin üzerine yazmaya devam edelim
%ayrıca satır ve sütunlar saklansın

[r,c]=size(imgD);
%görüntünün boyutunu değişkenlere atıyoruz

%bir faktör var başlangıç için bir alıyoruz sonra for ile satırlar boyunca
%döneceğiz satır sayısı j için 1'e eşit. sütunların değeri her satırdan
%önce geliyor ve görselde ilerliyoruz sütuna doğru bu yüzden satır bazında
%çalışıyoruz ve değeri atıyoruz
factor=1;
for i =1:r
   for j=1:c
     x(i,j)=factor*log(1+imgD(i,j));
   end
end
```

```
subplot(1,2,1); imshow(imgD);title('önce');
subplot(1,2,2); imshow(x); title('sonra');
```

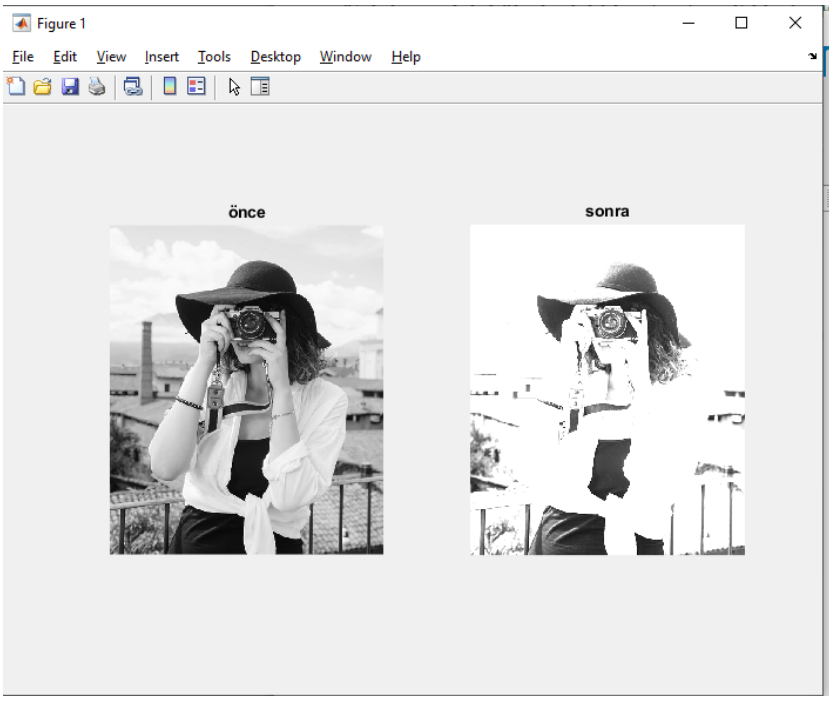
%daha parlak bölgelerdeki kontrast azaltıldı şimdi tersini yapalım

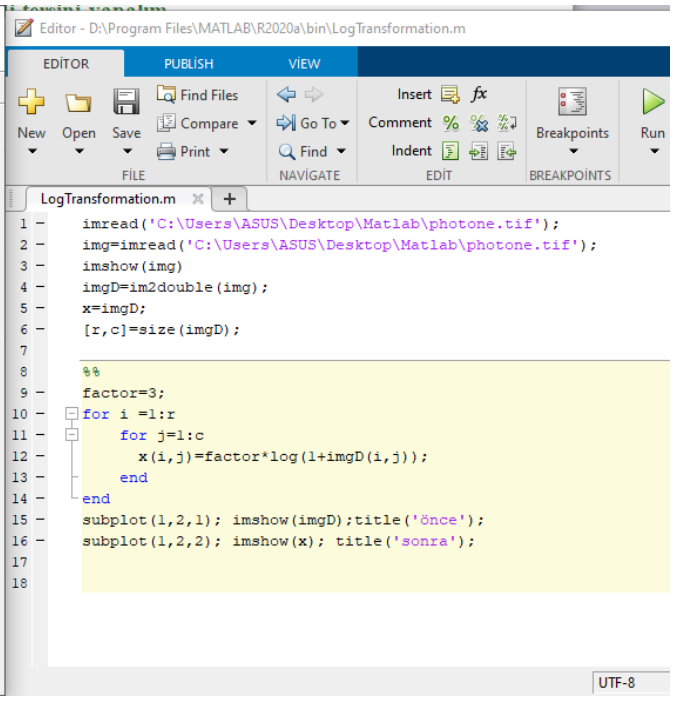
factor=6; olarak değiştiriyoruz sonuç=



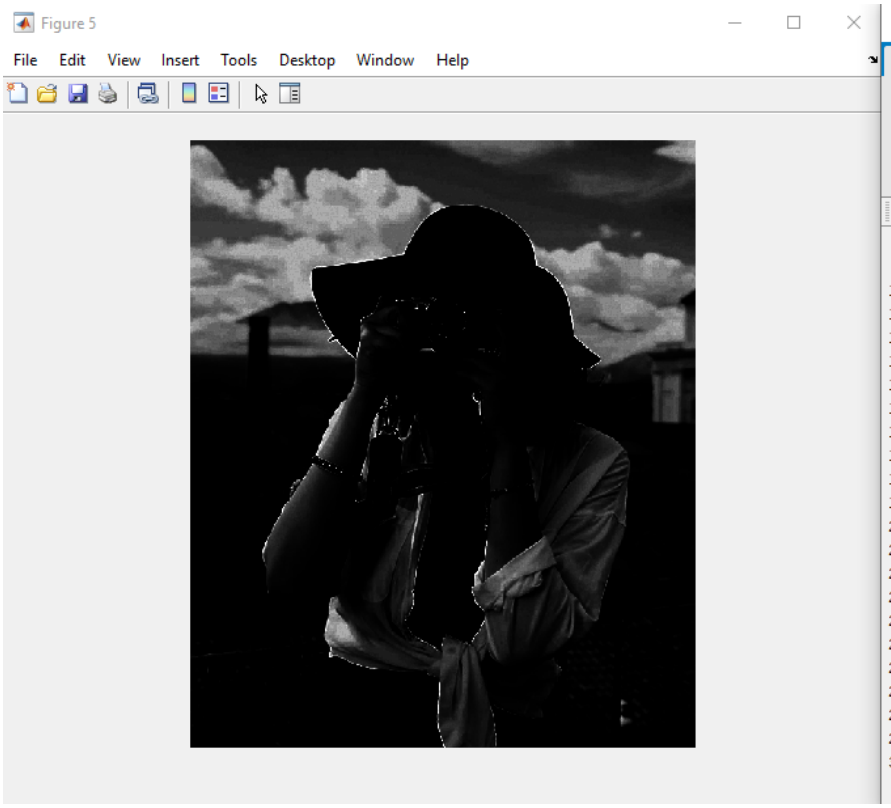


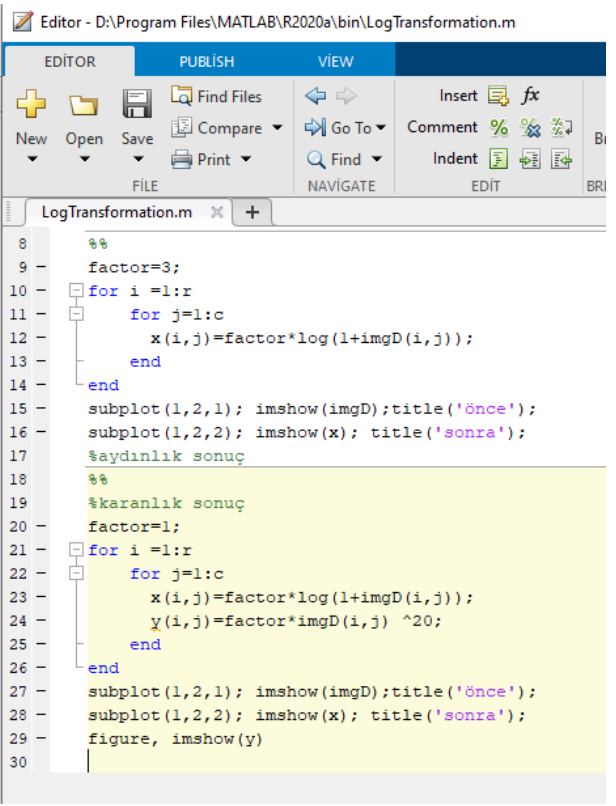
factor=3; olarak değiştiriyoruz sonuç=





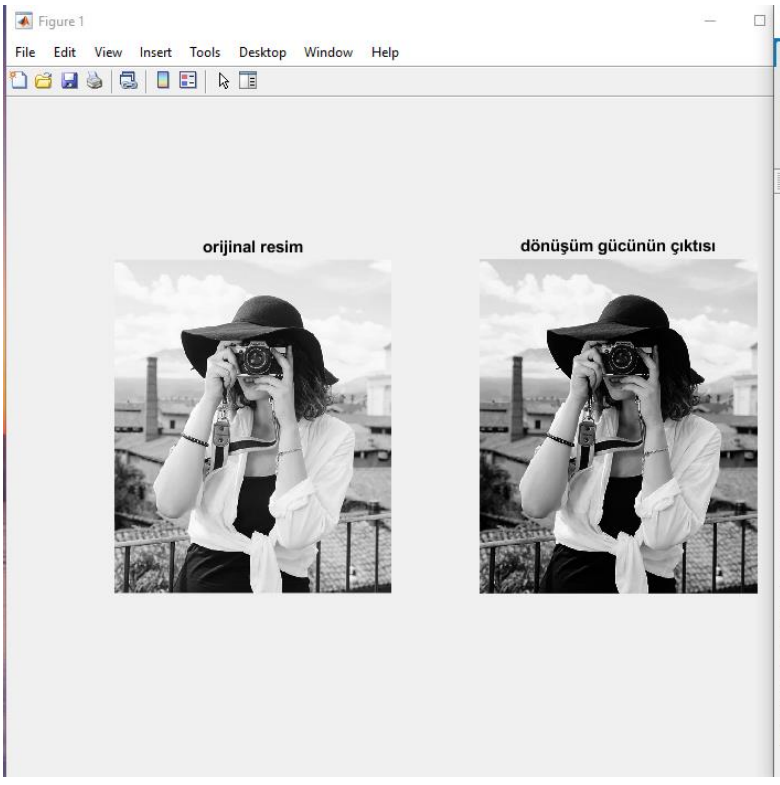
%güç işlevini kullanıyor gama işlevini değerleri değiştirelim karanlık görüntü elde edelim

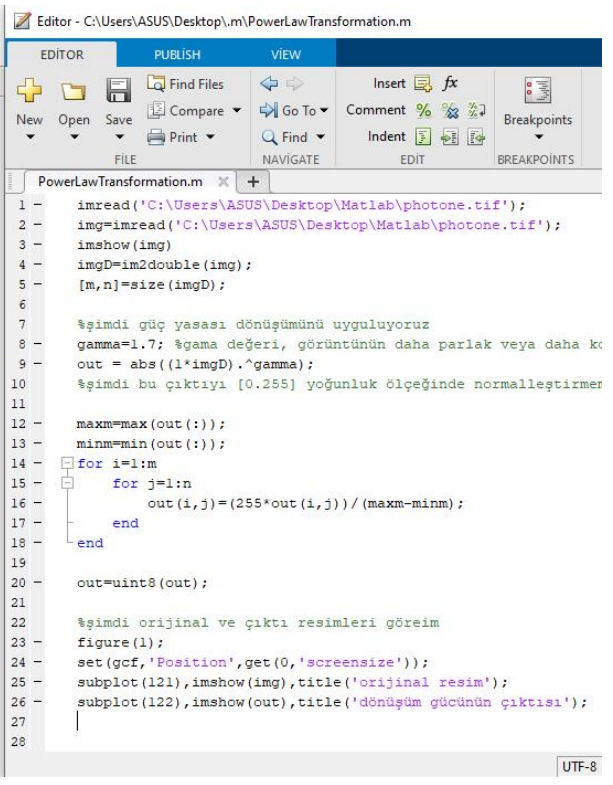





```
factor=1;
for i =1:r
    for j=1:c
      x(i,j)=factor*log(1+imgD(i,j));
      y(i,j)=factor*imgD(i,j) ^20;
    end
end
subplot(1,2,1); imshow(imgD);title('önce');
subplot(1,2,2); imshow(x); title('sonra');
figure, imshow(y)
```

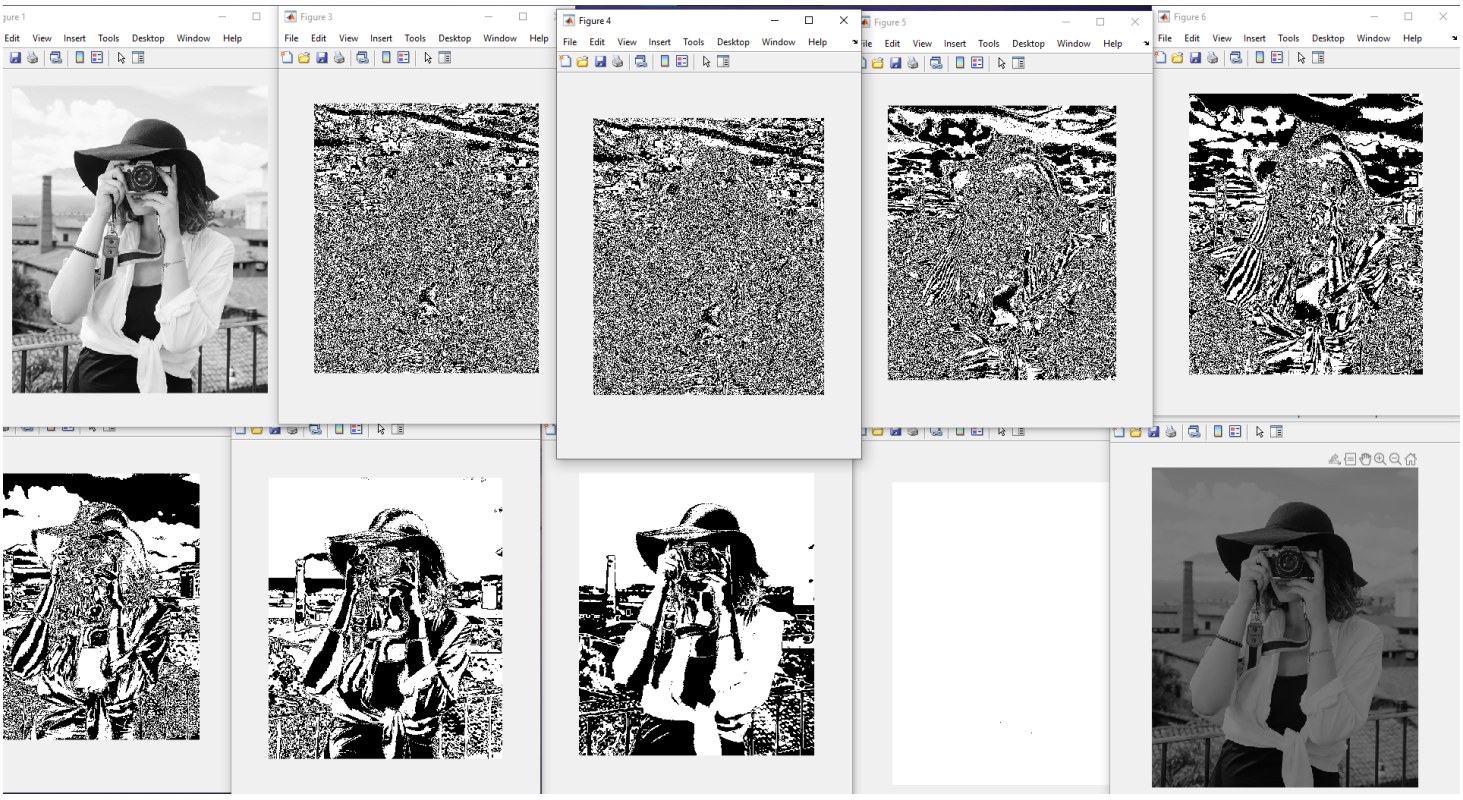
----------Power Law (Gamma) Transformation----------

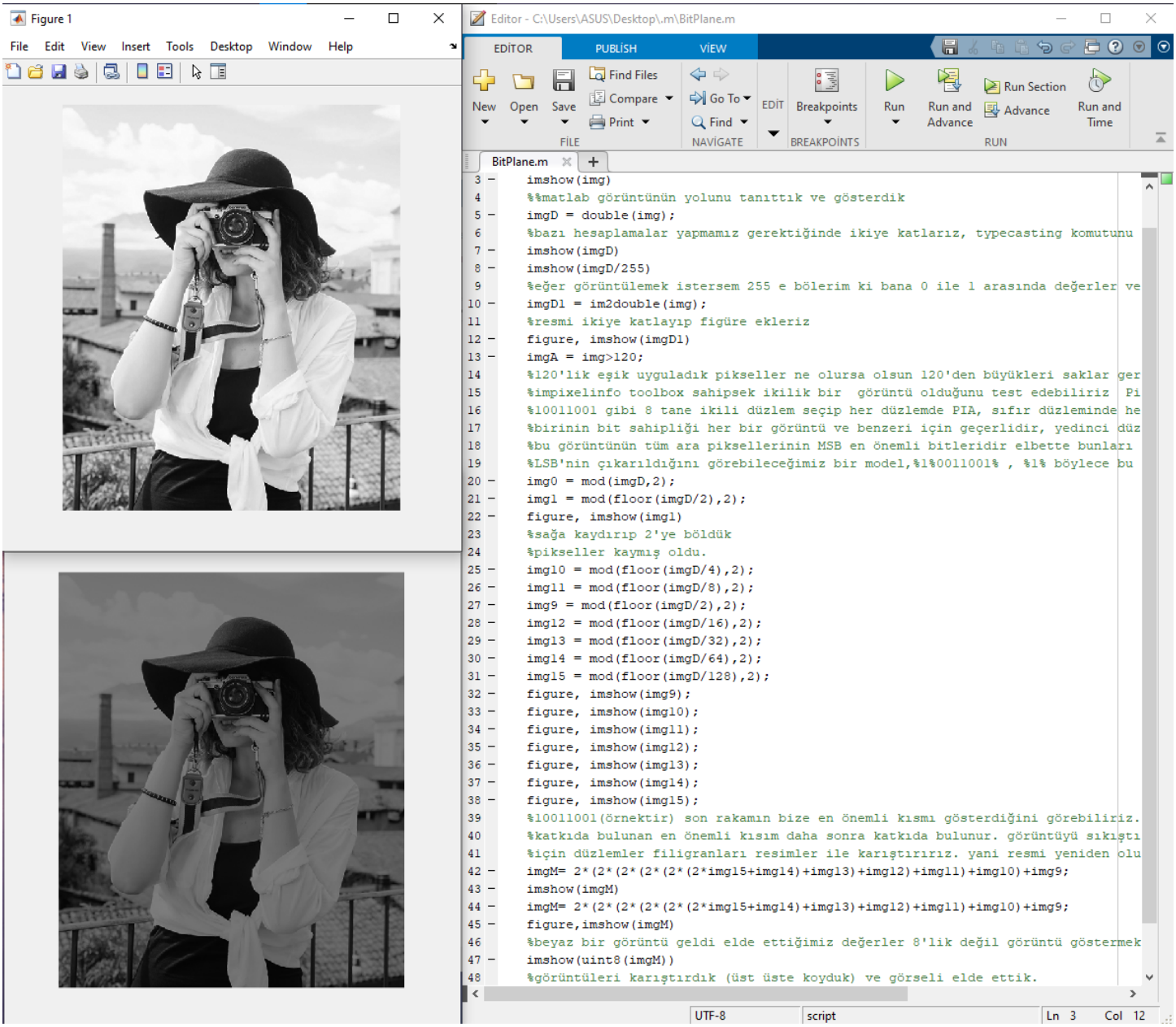




```
imread('C:\Users\ASUS\Desktop\Matlab\photone.tif');
img=imread('C:\Users\ASUS\Desktop\Matlab\photone.tif');
imshow(img)
imgD=im2double(img);
[m,n]=size(imgD);

%şimdi güç yasası dönüşümünü uyguluyoruz
gamma=1.7; %gama değeri, görüntünün daha parlak veya daha koyu olmasına bağlıdır
out = abs((1*imgD).^gamma);
%şimdi bu çıktıyı [0.255] yoğunluk ölçeğinde normalleştirmemiz gerekiyor

maxm=max(out(:));
minm=min(out(:));
for i=1:m
   for j=1:n
      out(i,j)=(255*out(i,j))/(maxm-minm);
   end
end

out=uint8(out);

%şimdi orijinal ve çıktı resimleri göreim
figure(1);
set(gcf,'Position',get(0,'screensize'));
subplot(121),imshow(img),title('orijinal resim');
subplot(122),imshow(out),title('dönüşüm gücünün çıktısı');
```

----------***Bit Plane***----------






```
imread('C:\Users\ASUS\Desktop\Matlab\photone.tif');
img=imread('C:\Users\ASUS\Desktop\Matlab\photone.tif');
imshow(img)
%%matlab görüntünün yolunu tanıttık ve gösterdik

imgD = double(img);
%bazı hesaplamalar yapmamız gerektiğinde ikiye katlarız, typecasting komutunu kullanabiliriz

imshow(imgD)
imshow(imgD/255)
%eğer görüntülemek istersem 255 e bölerim ki bana 0 ile 1 arasında değerler versin

imgD1 = im2double(img);
%resmi ikiye katlayıp figüre ekleriz

figure, imshow(imgD1)
imgA = img>120;
```

%120'lik eşik uyguladık pikseller ne olursa olsun 120'den büyükleri saklar geri kalanını sıfır olarak görebiliriz
%impixelinfo toolbox sahipsek ikilik bir görüntü olduğunu test edebiliriz Pixel info:(103,117) 0 gibi
%10011001 gibi 8 tane ikili düzlem seçip her düzlemde PIA, sıfır düzleminde her piksel için bit sıfıra sahiptir

%birinin bit sahipliği her bir görüntü ve benzeri için geçerlidir, yedinci düzlem en yüksek olana sahip olur ve
%bu görüntünün tüm ara piksellerinin MSB en önemli bitleridir elbette bunları elde etmek için dönüştürmeliyiz
%LSB'nin çıkarıldığını görebileceğimiz bir model,%1%0011001% , %1% böylece bu fon için bir kullanabiliriz.

```
img0 = mod(imgD,2);
img1 = mod(floor(imgD/2),2);
figure, imshow(img1)
```
%sağa kaydırıp 2'ye böldük
%pikseller kaymış oldu.

```
img10 = mod(floor(imgD/4),2);
img11 = mod(floor(imgD/8),2);
img9 = mod(floor(imgD/2),2);
img12 = mod(floor(imgD/16),2);
img13 = mod(floor(imgD/32),2);
img14 = mod(floor(imgD/64),2);
img15 = mod(floor(imgD/128),2);

figure, imshow(img9);
figure, imshow(img10);
figure, imshow(img11);
figure, imshow(img12);
figure, imshow(img13);
figure, imshow(img14);
figure, imshow(img15);
```

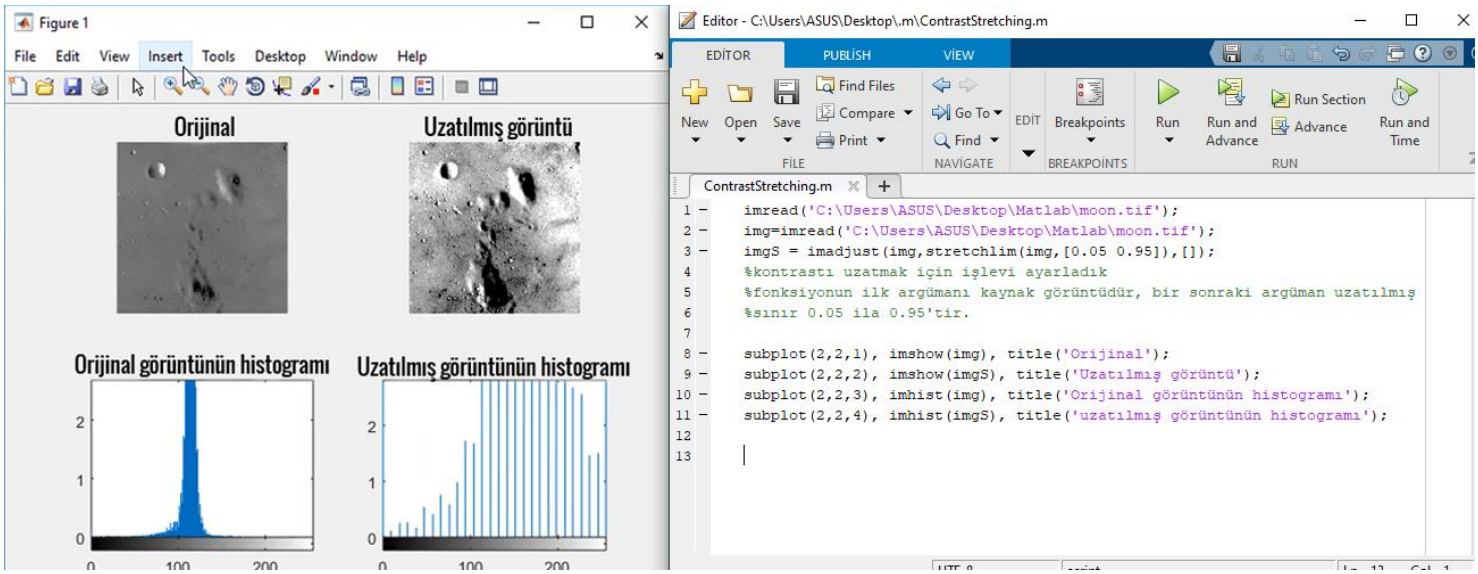
%10011001(örnektir) son rakamın bize en önemli kısmı gösterdiğini görebiliriz. son görüntünün maksimum değerine
%katkıda bulunan en önemli kısım daha sonra katkıda bulunur. görüntüyü sıkıştırmak veya başka bir görüntüyü depolamak
%için düzlemler filigranları resimler ile karıştırırız. yani resmi yeniden oluşturabiliriz yani birleşitiririz.

```
imgM= 2*(2*(2*(2*(2*(2*img15+img14)+img13)+img12)+img11)+img10)+img9;
imshow(imgM)
imgM= 2*(2*(2*(2*(2*(2*img15+img14)+img13)+img12)+img11)+img10)+img9;
figure,imshow(imgM)
```
%beyaz bir görüntü geldi elde ettiğimiz değerler 8'lik değil görüntü göstermek için dönüştürmek gerekiyor

```
imshow(uint8(imgM))
```
%görüntüleri karıştırdık (üst üste koyduk) ve görseli elde ettik.

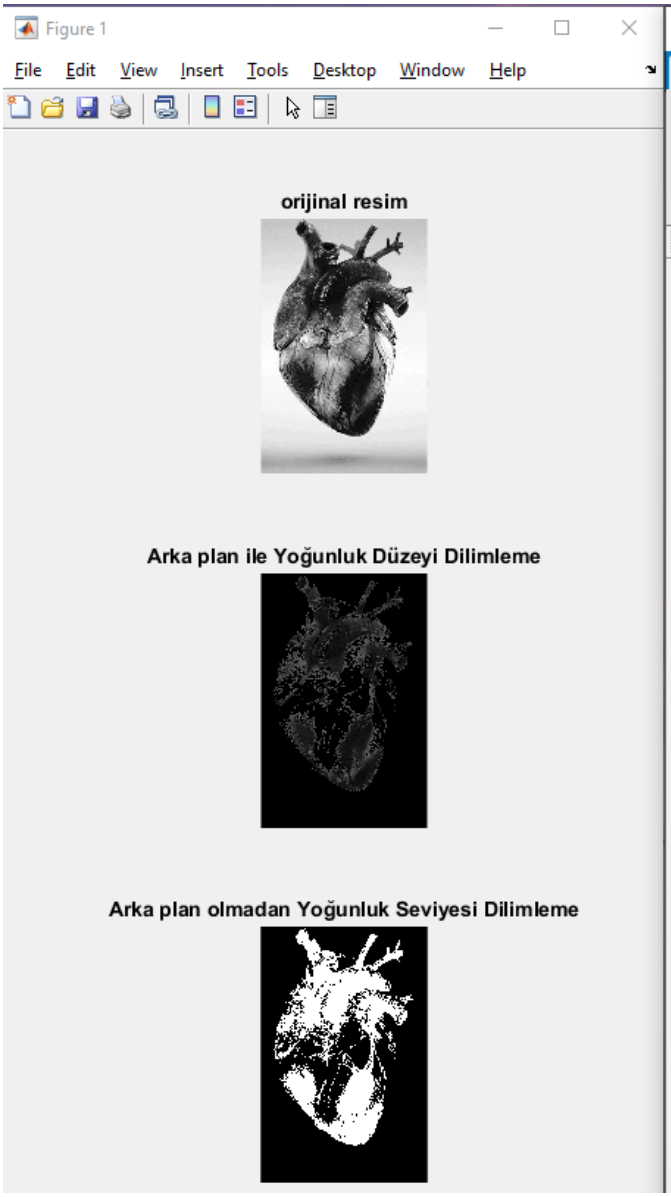
---------- *Piecewise-Linear Transformation Functions*----------

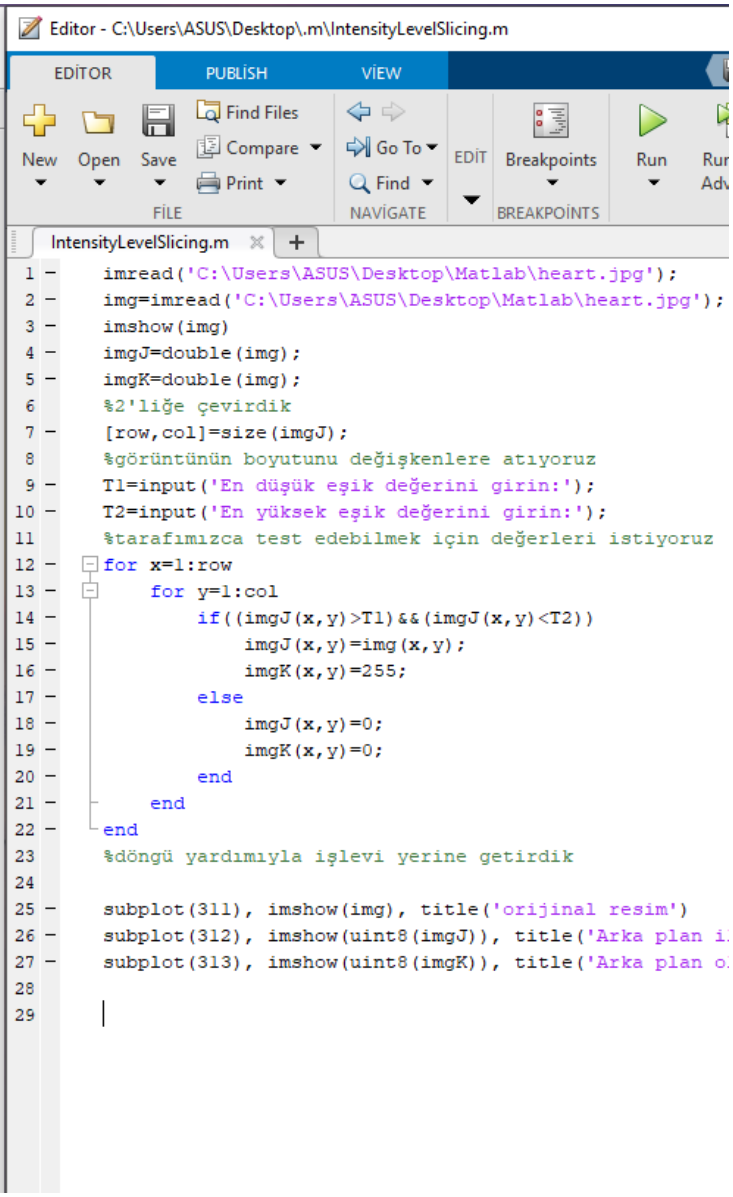
Contrast Stretching:




```
imread('C:\Users\ASUS\Desktop\Matlab\moon.tif');
img=imread('C:\Users\ASUS\Desktop\Matlab\moon.tif');
imgS = imadjust(img,stretchlim(img,[0.05 0.95]),[]);
%kontrastı uzatmak için işlevi ayarladık
%fonksiyonun ilk argümanı kaynak görüntüdür, bir sonraki argüman uzatılmış
%sınır 0.05 ila 0.95'tir.

subplot(2,2,1), imshow(img), title('Orijinal');
subplot(2,2,2), imshow(imgS), title('Uzatılmış görüntü');
subplot(2,2,3), imhist(img), title('Orijinal görüntünün histogramı');
subplot(2,2,4), imhist(imgS), title('uzatılmış görüntünün histogramı');
```

Intensity-Level Slicing (Gray Level Slicing):






```
imread('C:\Users\ASUS\Desktop\Matlab\heart.jpg');
img=imread('C:\Users\ASUS\Desktop\Matlab\heart.jpg');
imshow(img)
imgJ=double(img);
imgK=double(img);
%2'liğe çevirdik
[row,col]=size(imgJ);
%görüntünün boyutunu değişkenlere atıyoruz
T1=input('En düşük eşik değerini girin:');
T2=input('En yüksek eşik değerini girin:');
%tarafımızca test edebilmek için değerleri istiyoruz
```

```
for x=1:row
   for y=1:col
      if((imgJ(x,y)>T1)&&(imgJ(x,y)<T2))
         imgJ(x,y)=img(x,y);
         imgK(x,y)=255;
      else
         imgJ(x,y)=0;
         imgK(x,y)=0;
      end
   end
end
%döngü yardımıyla işlevi yerine getirdik

subplot(311), imshow(img), title('orijinal resim')
subplot(312), imshow(uint8(imgJ)), title('Arka plan ile Yoğunluk Düzeyi Dilimleme')
subplot(313), imshow(uint8(imgK)), title('Arka plan olmadan Yoğunluk Seviyesi Dilimleme')
```

BERK BAYRAKTARGİL 16008117058