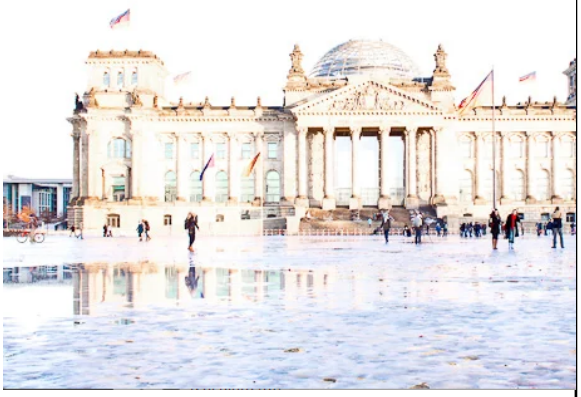
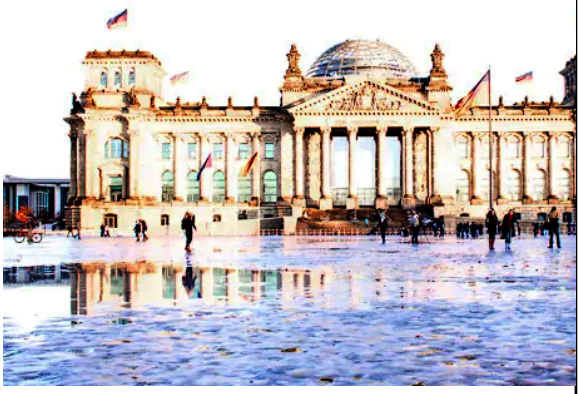
Serkan CAN 170220852

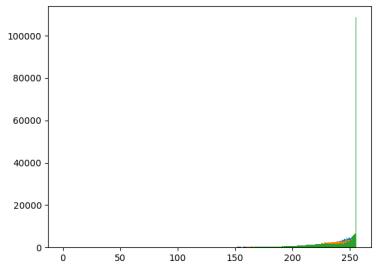
Geometrik Transformasyon ve Yeğinlik Dönüşüm Örnekleri	Bu alan kodları yazınız

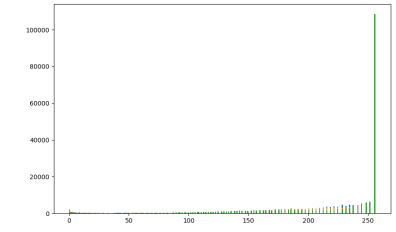
ilk



son

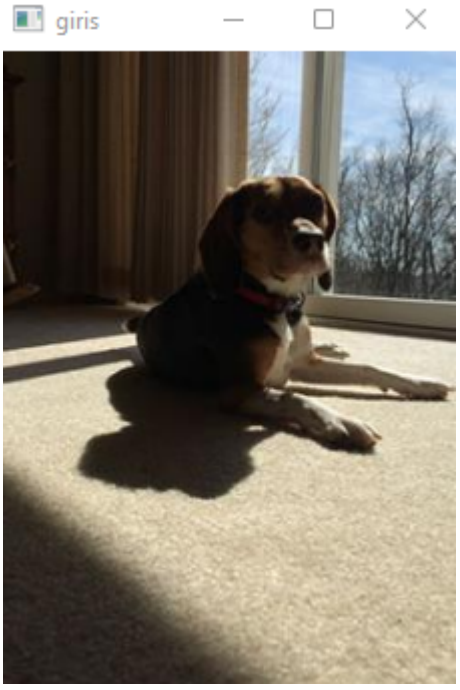


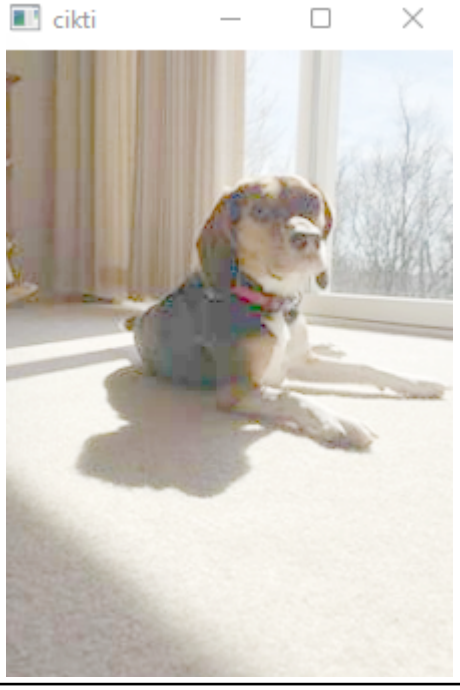




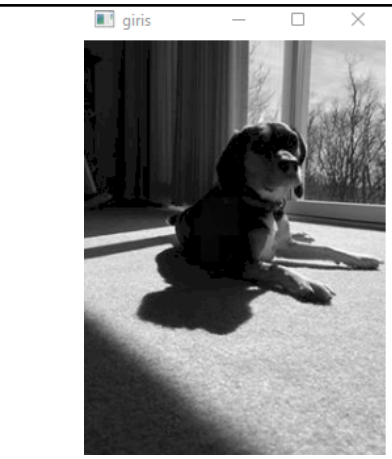
```
import cv2
import matplotlib.pyplot as plt
import numpy as np

def rescale(img):dizinin değerlerinin tekrar 255
e kadar dağılmasını sağlıyoruz

    s=img.astype(float)
    s-=np.min(s)
    s/=np.max(s)
    return (s*255).astype(np.uint8)

def gammaDonusum(r,c,gamma):
    r=r.astype(float)
    s=rescale(c*r**gamma)# cr^γ
    return s

img= cv2.imread("asiripozlanmis.png")
b,g,r=cv2.split(img)
cv2.imshow("ilk",img)
donusmus=gammaDonusum(img,1,gamma=3.5)

cv2.imshow("son",donusmus)
b2,g2,r2=cv2.split(donusmus)
plt.hist(b.ravel(),256,[0,256])
plt.hist(g.ravel(),256,[0,256])
plt.hist(r.ravel(),256,[0,256])
plt.show()
plt.hist(b2.ravel(),256,[0,256])
plt.hist(g2.ravel(),256,[0,256])
plt.hist(r2.ravel(),256,[0,256])
plt.show()
cv2.waitKey(0)
```

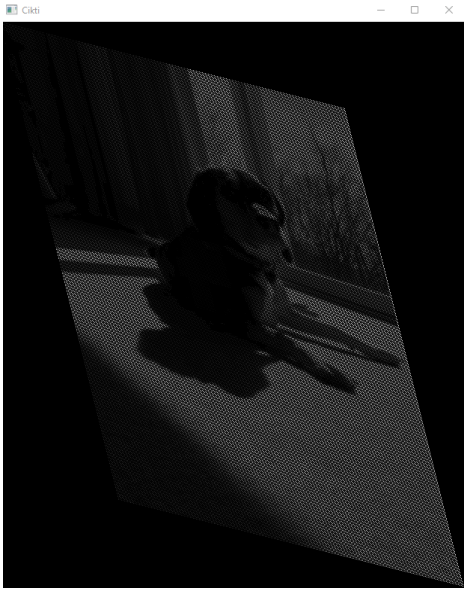
//gammaya 0-1 arasında değer verirsek, gamma dönüşüm grafiğindeki gibi siyahlar yayılır beyazlar sıkıştırılır. Bu resimdeki durumda beyaz bölgedeki sıkışmayı dağıtmak için 1 den büyük değer verip beyaz yeğinliğin geniş değer aralığına yayılmasını sağlayıp ideal değeri buluruz.





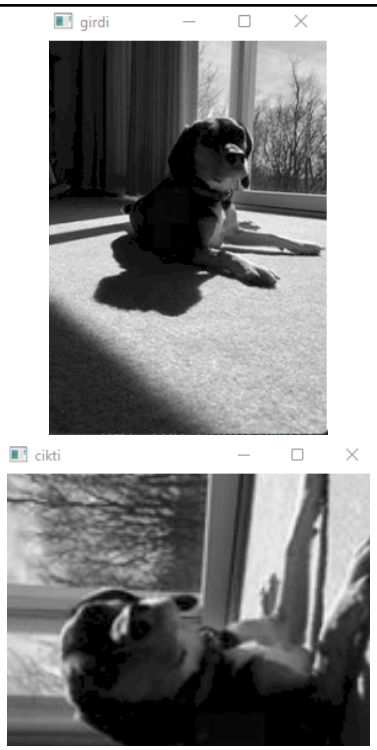

```python
import cv2
import matplotlib.pyplot as plt
import numpy as np

def rescale(img): #logaritmasını aldığımız matrisin değerlerinin tekrar 255 e kadar dağılmasını sağlıyoruz
    s=img.astype(float)
    s-=np.min(s)
    s/=np.max(s)
    return (s*255).astype(np.uint8)

def log_donustur(r,c):
    r=r.astype(float)
    log_img=c*np.log(1+r)#log1=0 oldugu icin değeri 0 olanı 1 yapmak amacıyla 1 ile topluyoruz
    return log_img.astype(np.uint8)


img= cv2.imread("dog.png")
cv2.imshow("giris",img)
cikti=log_donustur(img,12)
m=rescale(cikti)
print(np.max(m),np.min(m))
cv2.imshow("cikti",m)
cv2.waitKey(0)
```






```python
import cv2
import matplotlib.pyplot as plt
import numpy as np
def geotrans(I,sx,sy,shx,shy,tx,ty):
    x,y=I.shape #dizinin satır ve sütun sayısını buluyoruz.

T=np.array([[sx,shy,0],[shx,sy,0],[tx,ty,0]])
#lastik-levha dönüşümüne uygun matris oluşturuyoruz
    K=np.array([x,y,1])# çarpım uygulamak için satır, sütun sayısını dizi halinde yazıyoruz
    U=np.dot(K,T)#matris çarpımını yapıyoruz
    B=np.zeros((int(U[0]),int(U[1])))#Unun ilk sutunundaki kadar satır, ikinci sutunundaki kadar sütuna sahip 0lardan oluşan matris oluşturuyoruz

    for i in range (x-1):#indis 0 dan başladığı için -1
        for j in range (y-1):
            k=np.array([i,j,1])
            u=np.dot(k,T)#satır sütunları teker teker gezip lastik-levha formülünü uyguluyoruz, aynı matris sayısı fakat noktaların uzaklıkları değişir 1x3

B[int(u[0]),int(u[1])]=I[i,j].astype(np.uint8)
#lastik levha formülü uyguladığımız matrisin satır ve sütununa, orijinal resmin pikselindeki bilgiyi uint8 olarak yazıyoruz
            B=B.astype(np.uint8)
    return B


img=cv2.imread("dog.png",0)
cv2.imshow("giris",img)
sx=2;sy=2;shx=0.5;shy=0.5;tx=0;ty=0
B=geotrans(img,sx,sy,shx,shy,tx,ty)
cv2.imshow("Cikti",B)
cv2.waitKey(0)
```

//çıkışı büyüttüğümüz için arada piksel boşlukları kaldı, interpolasyon yöntemleri ile o boşlukları doldurabiliriz




```
import cv2
import matplotlib.pyplot as plt
import numpy as np

I=cv2.imread("dog.png",0)
x,y=I.shape#x,y piksel sayısını bulduk
center=(int(x/2),int(y/2))#merkezini bulduk
angle=90
scale=2
t=cv2.getRotationMatrix2D(center,angle,scale)


img=cv2.warpAffine(I,t,(x,y))
cv2.imshow("cikti",img)
cv2.imshow("girdi",I)
cv2.waitKey(0)
```


```
import cv2
import matplotlib.pyplot as plt
import numpy as np

def negatif_al(img):
    L=np.max(img)
    negatif=L-img #her pikselden çıkarma işlemi yapılır. Siyah değerler beyaza, beyaz değerler siyaha kayar.
    return negatif

img=cv2.imread("dog.png",0)
negatif=negatif_al(img)
cv2.imshow("giris",img)
print(img)
cv2.imshow("negatif",negatif)
print(negatif)
cv2.waitKey(0)
```