بسمه تعالی

تمرین سری چهارم پردازش تصاویر

مرضیه امیری

دانشجو ارشد هوش مصنوعی

تمرین سری ۴

۱) تصویر زیر را خوانده و قسمت قرمز را جدا کنید و بقیه تصویر را بصورت خاکستری نمایش دهید.

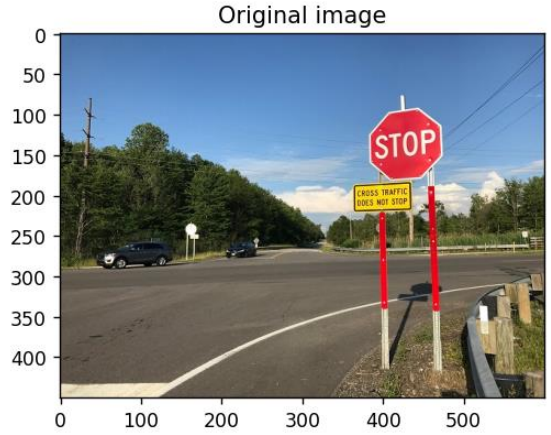
الف) اینکار را یکبار با روش مکعب و یکبار با روش اقلیدسی (کره) انجام دهید

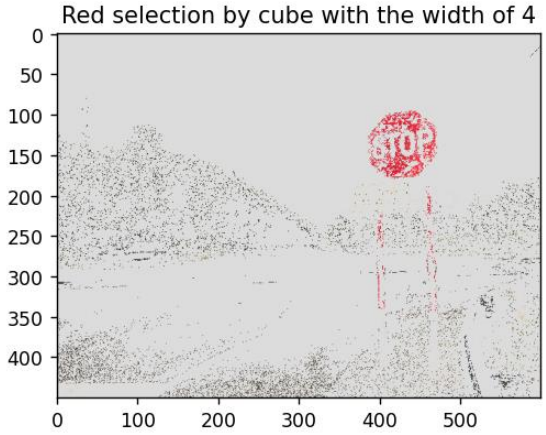
ب) میزان threshold برای جداسازی را با مقادیر مختلف تست کرده و نمایش دهید.

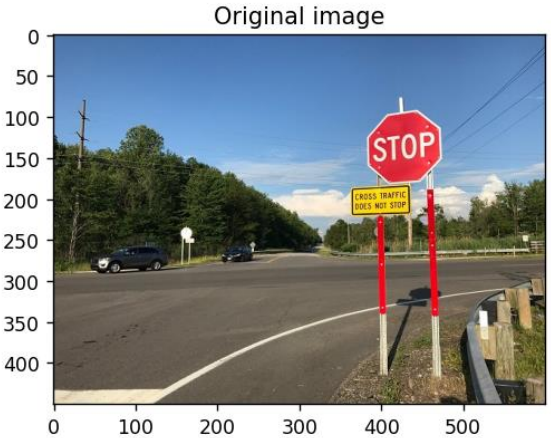
روش مکعب :

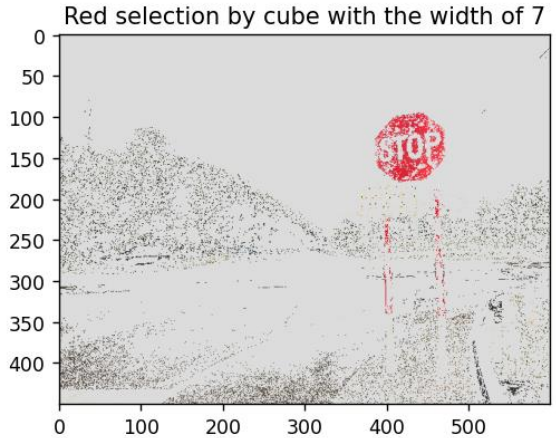
```
import cv2
import matplotlib.pyplot as plt
import numpy as np

#red selection function get image name and cutoff as the width of the cube
def Red_Selection(img1,cutoff):
    img = cv2.imread(img1,1) #read image
    imgcopy = cv2.imread(img1,1)
    h , w , ch = img.shape #image size
    cu = cutoff #width of the cube
    co = np.array([0,0,0])
    '''
    I chose the intensity of a red pixel by paint
    If the subtraction of that red pixel from the other pixels is
    larger than the half of the width of the cube
    make it [220,220,220]
    otherwise
    keep it
       '''
    for i in range(h):
        for j in range(w):
            co[2] = (220 - img[i][j][2]) #red value of pixel
            co[1] = (50 - img[i][j][1]) #green value of pixel
            co[0] = (70 - img[i][j][0]) #blue value of pixel
            if ((abs(co)) > [cu//2,cu//2,cu//2]).all() :
                img[i][j] = [220,220,220]

    #Show images
    f, subplt1 = plt.subplots(1,2,figsize=(10,10))
    subplt1[0].imshow(cv2.cvtColor(imgcopy, cv2.COLOR_BGR2RGB))
    subplt1[0].set_title("Original image")
    subplt1[1].imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    subplt1[1].set_title("Red selection by cube with the width of
{0}".format(cutoff))
    plt.show()
```

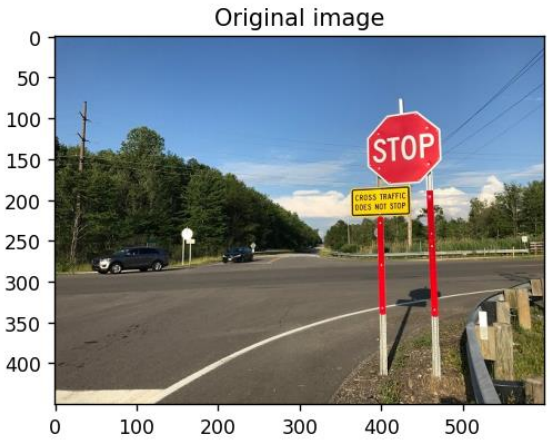
```
#execution
Red_Selection('1.png',4) #cutoff == 4
Red_Selection('1.png',7) #cutoff == 7
Red_Selection('1.png',10) #cutoff == 10
Red_Selection('1.png',20) #cutoff == 20
```

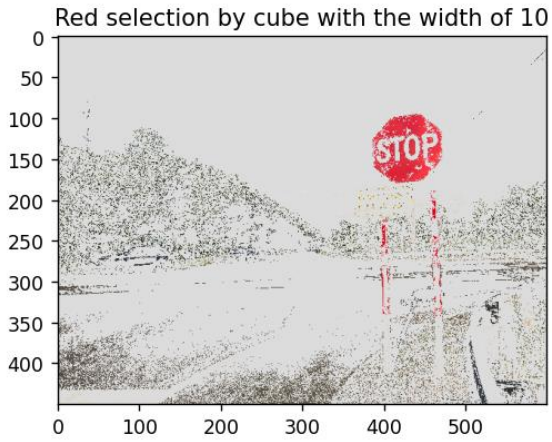


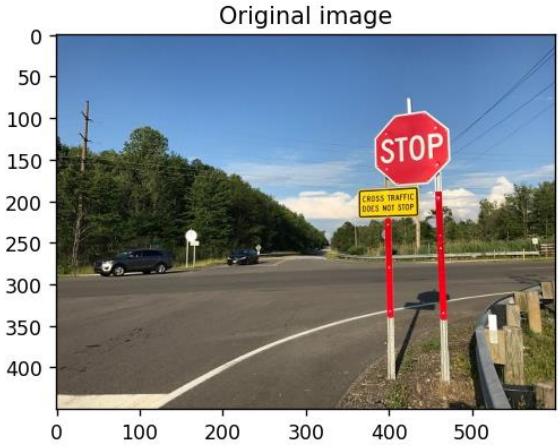


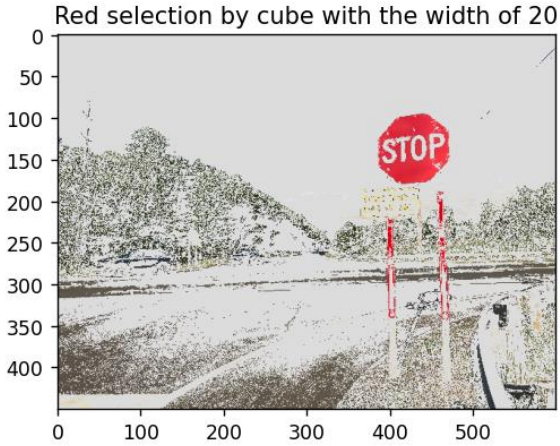








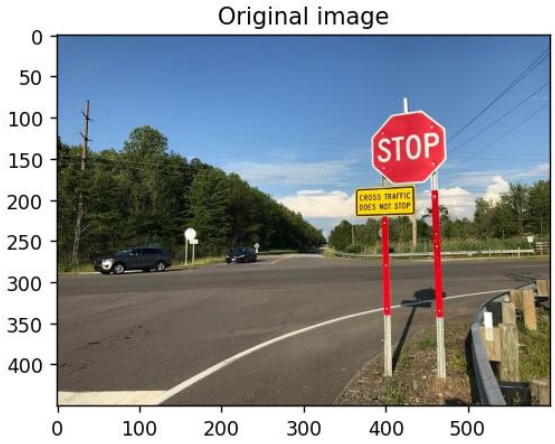


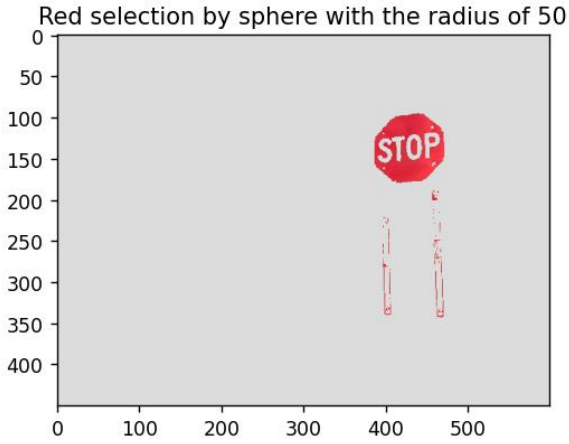


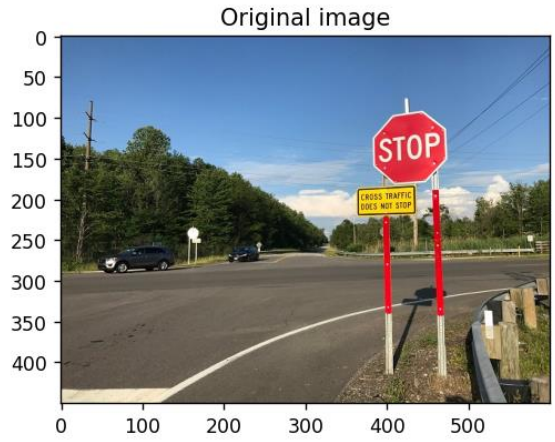
روش اقلیدسی (کره):

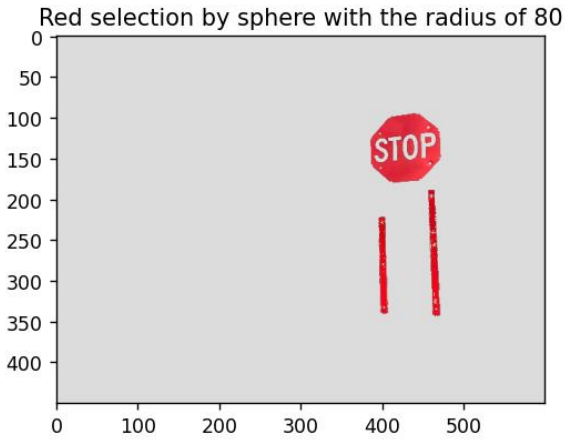
```
import cv2
import matplotlib.pyplot as plt
import numpy as np

#Red selection function get image name and cutoff as the radius of the sphere
def Red_Selection(img1,cutoff):
    img = cv2.imread(img1,1) #read image
    imgcopy = cv2.imread(img1,1)

    h , w , ch = img.shape #image size
    r = cutoff #radius of the image
```

```
'''
    I chose the intensity of a red pixel by paint
    If the distance of other pixels from the red pixel is
    larger than the radius of the sphere
    make it [220,220,220]
    otherwise
    keep it
    '''
    for i in range(h):
        for j in range(w):
            co = (220 - img[i][j][2]) **2 + (50 - img[i][j][1])**2 + (70 -
img[i][j][0])**2
            if ((co) > [r**2,r**2,r**2]).all() :
                img[i][j] = [220,220,220]

    #Show images
    f, subplt1 = plt.subplots(1,2,figsize=(10,10))
    subplt1[0].imshow(cv2.cvtColor(imgcopy, cv2.COLOR_BGR2RGB))
    subplt1[0].set_title("Original image")
    subplt1[1].imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    subplt1[1].set_title("Red selection by sphere with the width of
{0}".format(cutoff))
    plt.show()

Red_Selection('1.png',50) #cutoff == 50
Red_Selection('1.png',80) #cutoff == 80
Red_Selection('1.png',100) #cutoff == 100
Red_Selection('1.png',120) #cutoff == 120
```


Original image
STOP
CROSS TRAFFIC
DOES NOT STOP


Red selection by sphere with the radius of 50
STOP


Original image
STOP
CROSS TRAFFIC
DOES NOT STOP


Red selection by sphere with the radius of 80
STOP

ج) آیا میتوان این مقدار را بصورت یک رابطه ریاضیاتی بیان کرد؟ چه نتیجه ای دارد؟

روش کره بهتر عمل میکند. در روش کره شعاع ۱۰۰ بهترین عملکرد را دارد ولی در روش مکعبی نتیجه شعاع ۲۰ نتیجه بهتری دارد ولی صفحه کاملا خاکستری نشده است.

روش مکعب :

If $|R_j - a_j| > w/2$ → set to gray
Otherwise R_j

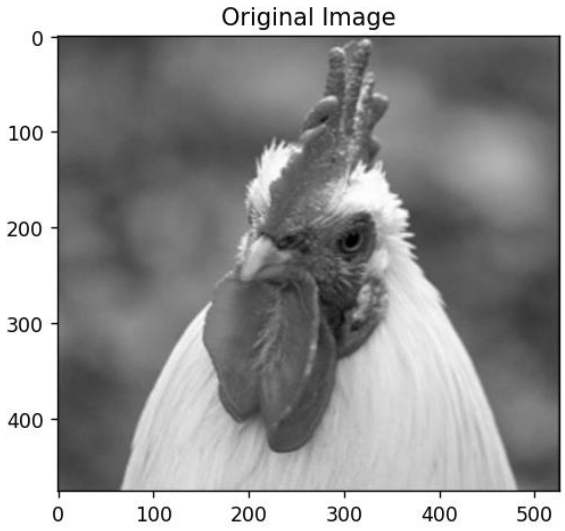
روش کره:

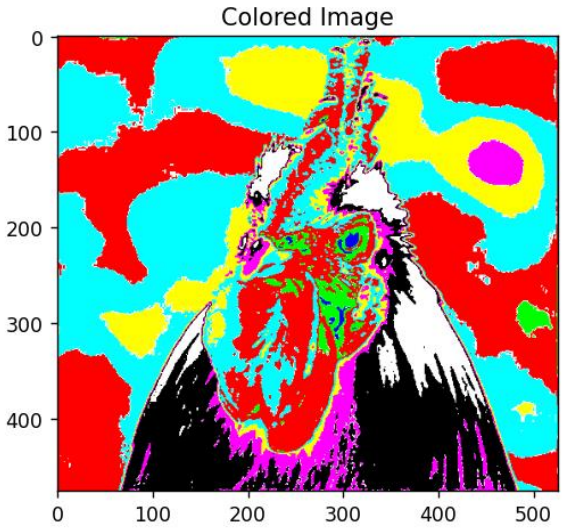
If $\sum_{j=1}^{j=n}(Rj - aj)^2 > R_0^2$ → set to gray
Otherwise Rj

۲) تصویر زیر را یک مرتبه با بخش بندی شدت و یک مرتبه با transformation رنگی کنید.(در روش دوم سعی کنید تابع را طوری تغییر دهید که تاج خروس قرمز شود)

بخش بندی شدت:

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

img = cv2.imread('2.png',0)
imgcopy = cv2.imread('2.png',0)
h , w = img.shape
imgcolored = np.zeros((h,w,3),np.uint8)
#divide the image intensities into 8 values
for i in range(h):
    for j in range(w):
        if  img[i][j] < 32:
            imgcolored[i][j] = [255,0,0]
        elif 32 < img[i][j] < 64:
            imgcolored[i][j] = [0,255,0]
        elif 64 < img[i][j] < 96:
            imgcolored[i][j] = [0,0,255]
        elif 96 < img[i][j] < 128:
            imgcolored[i][j] = [255,255,0]
        elif 128 < img[i][j] < 160:
            imgcolored[i][j] = [0,255,255]
        elif 160 < img[i][j] < 192:
            imgcolored[i][j] = [255,0,255]
        elif 192 < img[i][j] < 224:
            imgcolored[i][j] = [0,0,0]
        else:
            imgcolored[i][j] = [255,255,255]


#Show images
f, subplt1 = plt.subplots(1,2,figsize=(10,10))
subplt1[0].imshow(cv2.cvtColor(imgcopy, cv2.COLOR_BGR2RGB))
subplt1[0].set_title("Original Image")
subplt1[1].imshow(cv2.cvtColor(imgcolored, cv2.COLOR_BGR2RGB))
subplt1[1].set_title("Colored Image")
plt.show()
```



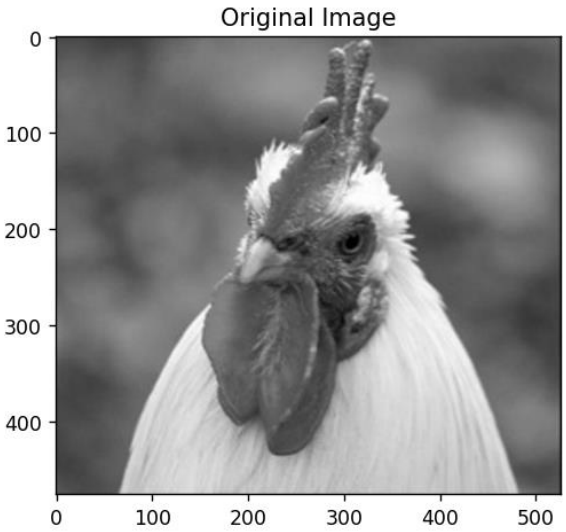


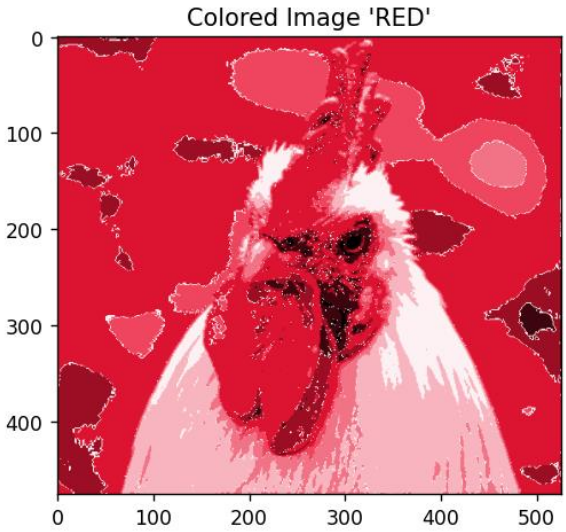
Transformation :

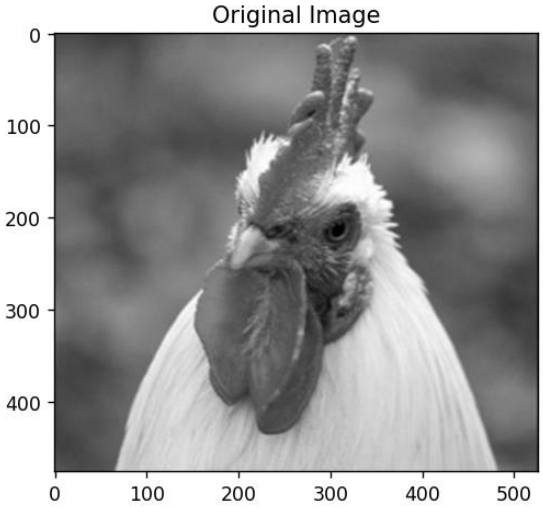
```
import numpy as np
import cv2
import matplotlib.pyplot as plt

img = cv2.imread('2.png',0)
imgcopy = cv2.imread('2.png',0)

h , w = img.shape
#RED
imgcoloredR = np.zeros((h,w,3),np.uint8)
for i in range(h):
    for j in range(w):
        if  img[i][j] < 32:
                imgcoloredR[i][j] = [0,0,0]
        elif 32 < img[i][j] < 64:
            imgcoloredR[i][j] = [14,5,61]
        elif 64 < img[i][j] < 80:
            imgcoloredR[i][j] = [35,14,156]
        elif 80 < img[i][j] < 130:
            imgcoloredR[i][j] = [50,19,221]
        elif 130 < img[i][j] < 160:
            imgcoloredR[i][j] = [95,69,239]
        elif 160 < img[i][j] < 192:
            imgcoloredR[i][j] = [134,115,242]
```

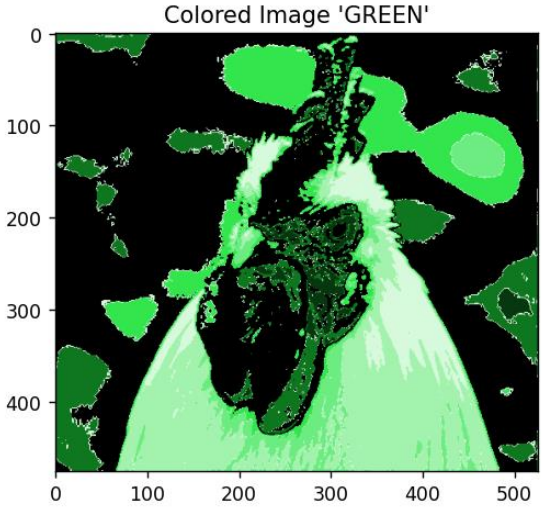
```
        elif 192 < img[i][j] < 224:
            imgcoloredR[i][j] = [191,181,249]
        else:
            imgcoloredR[i][j] = [243,241,254]

#Show images
f, subplt1 = plt.subplots(1,2,figsize=(10,10))
subplt1[0].imshow(cv2.cvtColor(imgcopy, cv2.COLOR_BGR2RGB))
subplt1[0].set_title("Original Image")
subplt1[1].imshow(cv2.cvtColor(imgcoloredR, cv2.COLOR_BGR2RGB))
subplt1[1].set_title("Colored Image 'RED'")
plt.show()

#GREEN
imgcoloredG = np.zeros((h,w,3),np.uint8)
for i in range(h):
    for j in range(w):
        if  img[i][j] < 32:
                imgcoloredG[i][j] = [0,0,0]
        elif 32 < img[i][j] < 64 :
            imgcoloredG[i][j] = [15,56,7]
        elif 64 < img[i][j] < 80:
            imgcoloredG[i][j] = [31,119,15]
        elif 80 < img[i][j] < 130:
            imgcoloredG[i][j] = [0,0,0]
        elif 130 < img[i][j] < 160:
            imgcoloredG[i][j] = [78,230,51]
        elif 160 < img[i][j] < 192:
            imgcoloredG[i][j] = [129,237,109]
        elif 192 < img[i][j] < 224:
            imgcoloredG[i][j] = [174,243,163]
        else:
            imgcoloredG[i][j] = [219,250,214]

#Show images
f, subplt1 = plt.subplots(1,2,figsize=(10,10))
subplt1[0].imshow(cv2.cvtColor(imgcopy, cv2.COLOR_BGR2RGB))
subplt1[0].set_title("Original Image")
subplt1[1].imshow(cv2.cvtColor(imgcoloredG, cv2.COLOR_BGR2RGB))
subplt1[1].set_title("Colored Image 'GREEN'")
plt.show()
```

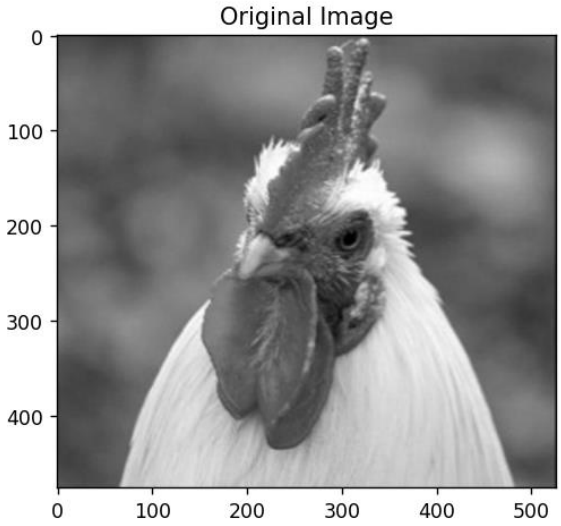
```
#BLUE
imgcoloredB = np.zeros((h,w,3),np.uint8)
for i in range(h):
    for j in range(w):
        if  img[i][j] < 32:
                imgcoloredB[i][j] = [3,1,16]
        elif 32 < img[i][j] < 64:
            imgcoloredB[i][j] = [72,6,18]
        elif 64 < img[i][j] < 80:
            imgcoloredB[i][j] = [137,12,34]
        elif 80 < img[i][j] < 130:
            imgcoloredG[i][j] = [0,0,0]
        elif 128 < img[i][j] < 160:
            imgcoloredB[i][j] = [238,47,81]
        elif 160 < img[i][j] < 192:
            imgcoloredB[i][j] = [243,114,136]
        elif 192 < img[i][j] < 224:
            imgcoloredB[i][j] = [250,179,191]
        else:
            imgcoloredG[i][j] = [252,220,225]

#Show images
f, subplt1 = plt.subplots(1,2,figsize=(10,10))
subplt1[0].imshow(cv2.cvtColor(imgcopy, cv2.COLOR_BGR2RGB))
subplt1[0].set_title("Original Image")
subplt1[1].imshow(cv2.cvtColor(imgcoloredB, cv2.COLOR_BGR2RGB))
subplt1[1].set_title("Colored Image 'BLUE'")
plt.show()

#addition
imagecolored_final = np.zeros((h,w,3),np.uint8)
for i in range(h):
    for j in range(w):
        imagecolored_final[i][j] =(imgcoloredG[i][j]+ imgcoloredB[i][j] +
imgcoloredR[i][j])

#Show images
f, subplt1 = plt.subplots(1,2,figsize=(10,10))
subplt1[0].imshow(cv2.cvtColor(imgcopy, cv2.COLOR_BGR2RGB))
subplt1[0].set_title("Original Image")
subplt1[1].imshow(cv2.cvtColor(imagecolored_final, cv2.COLOR_BGR2RGB))
subplt1[1].set_title("Colored Image 'Final'")
plt.show()
```

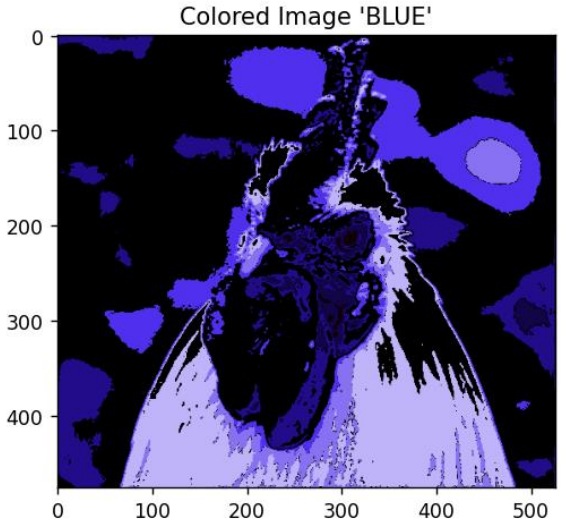

Original Image
0
100
200
300
400
0
100
200
300
400
500

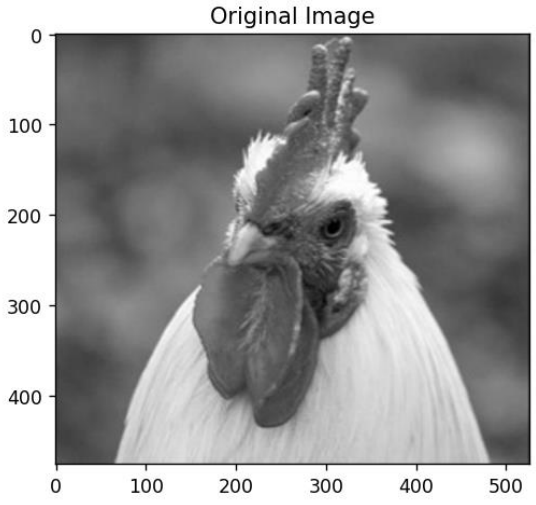

Colored Image 'RED'
0
100
200
300
400
0
100
200
300
400
500

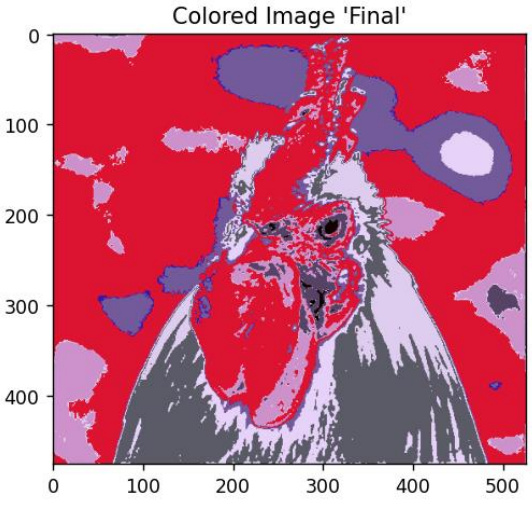

Original Image
0
100
200
300
400
0
100
200
300
400
500


Colored Image 'GREEN'
0
100
200
300
400
0
100
200
300
400
500


Original Image
0
100
200
300
400
0
100
200
300
400
500


Colored Image 'BLUE'
0
100
200
300
400
0
100
200
300
400
500


Original Image
0
100
200
300
400
0
100
200
300
400
500


Colored Image 'Final'
0
100
200
300
400
0
100
200
300
400
500