LAB 2 Exercises

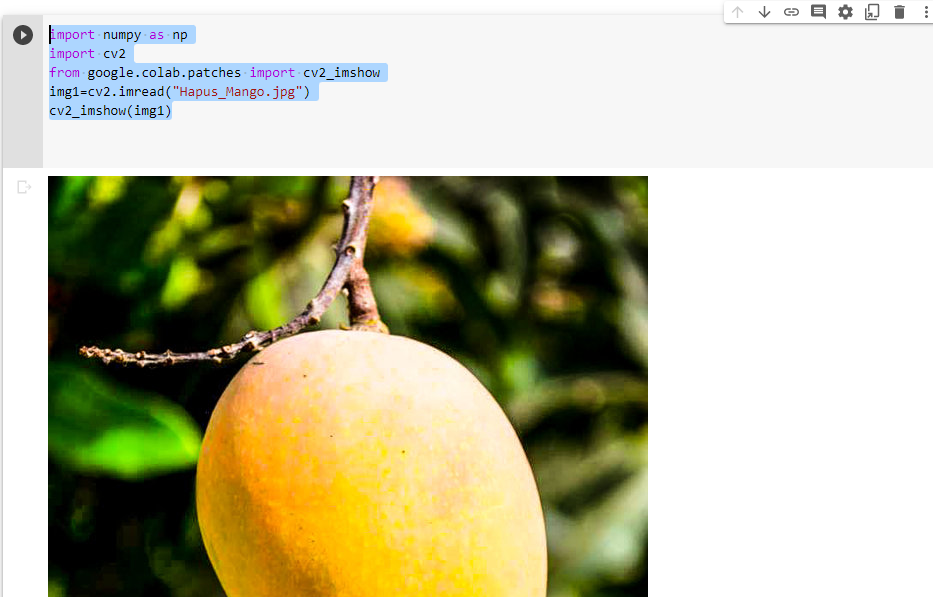
import numpy as np

import cv2

from google.colab.patches import cv2\_imshow

img1=cv2.imread("Hapus\_Mango.jpg")

cv2\_imshow(img1)



red=img1[:,:,0]

cv2\_imshow(red)

A picture containing text

Description automatically generated

green=img1[:,:,1]

cv2\_imshow(green)

A picture containing text

Description automatically generated

blue=img1[:,:,2]

cv2\_imshow(blue)

A picture containing text

Description automatically generated

plt.subplot(141)

plt.title("Original Image")

plt.xticks([]),plt.yticks([])

plt.imshow(img1)

plt.subplot(142)

plt.title("red")

plt.xticks([]),plt.yticks([])

plt.imshow(red)

Graphical user interface, application, Teams

Description automatically generated

plt.subplot(143)

plt.title("green")

plt.xticks([]),plt.yticks([])

plt.imshow(green)

plt.subplot(144)

plt.title("blue")

plt.xticks([]),plt.yticks([])

plt.imshow(blue)

Graphical user interface, text, application, Teams

Description automatically generated

import cv2

from google.colab.patches import cv2\_imshow

img2 =cv2.imread("Hapus\_Mango.jpg")

rgb\_image=cv2.cvtColor(img2,cv2.COLOR\_BGR2RGB)

cv2\_imshow(rgb\_image)



import cv2

import numpy as np

img2= cv2.imread("Hapus\_Mango.jpg")

img2.shape

cropped = img2[10:100,10:150]

cv2\_imshow(cropped)

Graphical user interface, application

Description automatically generated

import numpy as np

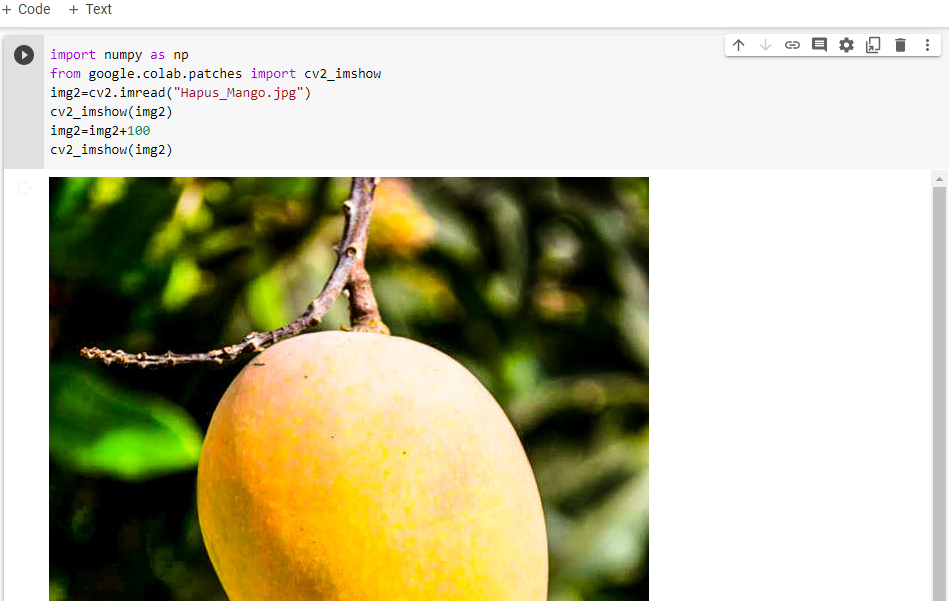
from google.colab.patches import cv2\_imshow

img2=cv2.imread("Hapus\_Mango.jpg")

cv2\_imshow(img2)

img2=img2+100

cv2\_imshow(img2)



A picture containing chart

Description automatically generated

import numpy as np

import cv2

from google.colab.patches import cv2\_imshow

img1=cv2.imread("blackandwhite.jpg")

cv2\_imshow(img1)

neg=255-img1

cv2\_imshow(neg)

Graphical user interface

Description automatically generated

A picture containing text, outdoor, shore

Description automatically generated

import numpy as np

import cv2

from google.colab.patches import cv2\_imshow

img=cv2.imread("blackandwhite.jpg")

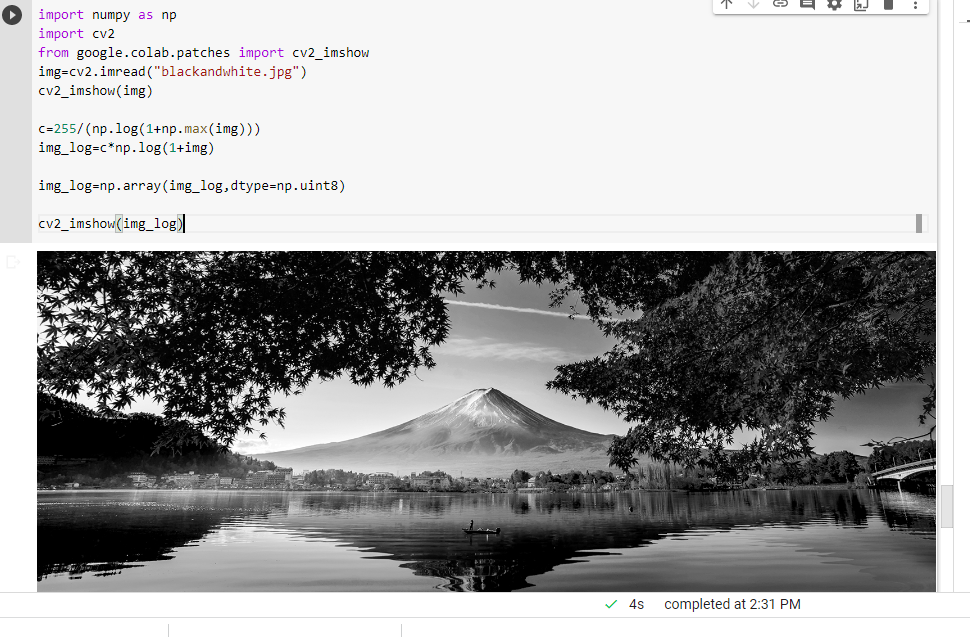
cv2\_imshow(img)

c=255/(np.log(1+np.max(img)))

img\_log=c\*np.log(1+img)

img\_log=np.array(img\_log,dtype=np.uint8)

cv2\_imshow(img\_log)



A lake with trees and mountains in the background

Description automatically generated with low confidence

import numpy as np

import cv2

from google.colab.patches import cv2\_imshow

img1=cv2.imread("blackandwhite.jpg")

cv2\_imshow(img1)

im1=np.array(255\*(img1/255) \*\*0.6,dtype='uint8')

cv2\_imshow(im1)



A lake with trees and mountains in the background

Description automatically generated with medium confidence

height, width,\_=img2.shape

for i in range(0,height-1):

  for j in range(0,width-1):

    pixel=img2[i,j]

    #Negate each channel by subtrating it from 255

    #1st index contains red pixel

    pixel[0]=255-pixel[0]

    #2nd index contains green pixel

    pixel[1]=255-pixel[1]

    #3rd index contains blue pixel

    pixel[2]=255-pixel[2]

    #store new values in the pixel

    img2[i,j]=pixel

#Display the negative transformed image

plt.imshow(img2)

plt.show()

Graphical user interface, text, application

Description automatically generated

import cv2

from google.colab.patches import cv2\_imshow

img=cv2.imread("card.jpg",0)

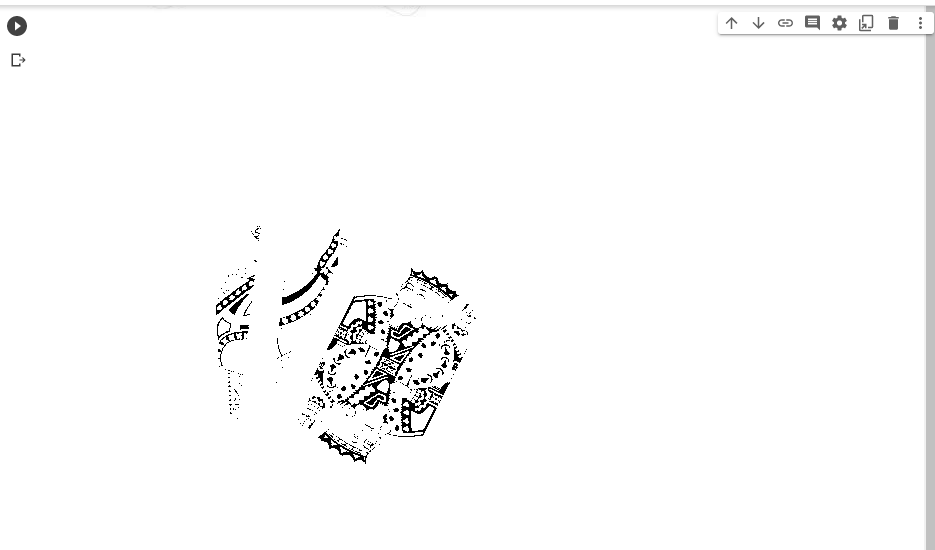
retval,threshold=cv2.threshold(img,62,255,cv2.THRESH\_BINARY)

cv2\_imshow( img)

cv2\_imshow(threshold)

Text

Description automatically generated



thresh=127

maxValue=255

th,dst=cv2.threshold(img,thresh,maxValue,cv2.THRESH\_TRUNC);

plt.imshow(dst,cmap='gray')

plt.axis('off')

plt.show()

Graphical user interface, text, application, email

Description automatically generated

image=cv2.imread("light.jfif")

#to convert the image in greyscale

img=cv2.cvtColor(image,cv2.COLOR\_BGR2GRAY)

ret,th1=cv2.threshold(img,160,255,cv2.THRESH\_BINARY)

th2=cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE\_THRESH\_MEAN\_C,cv2.THRESH\_BINARY,11,2)

th3=cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C,cv2.THRESH\_BINARY,11,2)

plt.subplot(141)

plt.title("Original Image")

plt.xticks([]),plt.yticks([])

plt.imshow(img,cmap='gray')

plt.subplot(142)

plt.title("binary")

plt.xticks([]),plt.yticks([])

plt.imshow(th1,cmap='gray')

plt.subplot(143)

plt.title("Mean Image")

plt.xticks([]),plt.yticks([])

plt.imshow(th2,cmap='gray')

plt.subplot(144)

plt.title("Gaussian Image")

plt.xticks([]),plt.yticks([])

plt.imshow(th3,cmap='gray')

Graphical user interface, text, application

Description automatically generated