import cv2 # Imports OpenCV package

import numpy as np # Imports Numpy package

import os # Imports python library

def generate\_image(): # This function will generate an image for first 5 tasks

im=np.ones(shape=(700,700,3)) \* 255 # Initialize a numpy array of size (700 x 700 x 3) filled with value = 255

im[100:300,400:500,1:]=0 # For blue rectangle, set corresponding pixel values in red and green channels to zero

im[30:400,100:250,0]=0 # For green rectangle, set corresponding pixel values in blue and red channels to zero

im[30:400,100:250,2]=0

im[500:600,300:600,0:2]=0 # For red rectangle, set corresponding pixel values in blue and green channels to zero

return im,im.shape # return image as numpy array and its shape

def write\_image(imagename,im): # This function will write image to the disk in output/[IMAGE\_NAME]

if(not os.path.exists('outputs')):

os.mkdir('outputs')

cv2.imwrite(os.path.join('outputs',imagename),im) # OpenCV function to write the image

''' You need to complete following function for task 1 '''

def count\_pixels(image,rows,cols,channels): # First task is to count the number of white, red, blue and total number of pixels in the image

whitecount=0 # Set white pixel counter to zero

redcount=0 # Set red pixel counter to zero

bluecount=0 # Set blue pixel counter to zero

totalcount=0 # Set counter of total number of pixels to zero

for k in range(channels): # Iterate through all channels

for r in range(rows): # Iterate through all pixel rows/height of the image

for c in range(cols): # Iterate through all pixel columns of the image

''' Write your code here'''

pass

print('White pixels:',whitecount) # Print final counts

print('Red Pixels:',redcount)

print('Blue Pixels:',bluecount)

print('Total Pixels:',totalcount)

''' You need to complete following function for task 2 '''

def red\_to\_blue(image,rows,cols,channels):

*''' Write your code here '''*

return image

''' You need to complete following function for task 3 '''

def remove\_blue(image,rows,cols,channels):

*''' Write your code here '''*

return image

''' You need to complete following function for task 4 '''

def green\_to\_black(image,rows,cols,channels):

*''' Write your code here '''*

return image

''' You need to complete following function for task 5 '''

def reduce\_brightness(image,rows,cols,channels):

*''' Write your code here '''*

return image

im,shape=generate\_image() # Generate the image

(rows,cols,channels)=shape # Assign shape values to proper variables

''' Call function for task 1 '''

count\_pixels(im.copy(),rows,cols,channels) # For task 1, this function will be called

''' Call function for task 2 '''

imcopy=red\_to\_blue(im.copy(),rows,cols,channels) # This function will be called for task 2. The function should return the output image

write\_image('red\_blue.jpg',imcopy) # Store the image returned by the function

''' Call function for task 3 '''

imcopy=remove\_blue(im.copy(),rows,cols,channels) # This function will be called for task 3. The function should return the output image

write\_image('blue\_removed.jpg',imcopy) # Store the image returned by the function

''' Call function for task 4 '''

imcopy= green\_to\_black(im.copy(),rows,cols,channels) # This function will be called for task 4. The function should return the output image

write\_image('green\_black.jpg',imcopy) # Store the image returned by the function

''' Call function for task 5 '''

imcopy=reduce\_brightness(im.copy(),rows,cols,channels) # This function will be called for task 5. The function should return the output image

write\_image('low.jpg',imcopy) # Store the image returned by the function

import cv2 # Import OpenCV package

import numpy as np # Imports numpy package

import os # Import a python library

''' You need to complete following function for task 6 '''

def binary\_threshold(image,rows,cols,channels): # This function will binary threshold the image

''' Enter your code here'''

return image # return the image

def read\_image(imagepath): # This function will read the image from imagepath

im=cv2.imread(imagepath) # OpenCV function to read the image as RGB

im=np.expand\_dims(im[:,:,0], axis=2) # Use only first (blue) channel of the image and reshape the array to have one channel (rows x columns x 1)

return im,im.shape

def write\_image(imagename,im): # This function will store image array (im) with name "imagename"

if(not os.path.exists('outputs')): # Create output directory if it doesn't exist already

os.mkdir('outputs')

cv2.imwrite(os.path.join('outputs',imagename),im) # Store the image in the path with the name "imagename"

im,shape=read\_image('circle.png') # Read circle.png image provided in the current directory

(rows,cols,channels)=shape

imcopy=binary\_threshold(im.copy(),rows,cols,channels) # Make a copy of the original array and pass it to the function

write\_image('threshcircle.jpg',imcopy) # Store the thresholded image. You can change the name of the image to run the code for different threshold values

import cv2

import numpy as np

im=cv2.imread('uh.png') # Reads image from current folder

height,width,channels=im.shape # Gets the shape of the image

print('Original Image Dimensions:',im.shape)

maxpooled=np.empty(shape=(height//2,width//2,3)) # Initializing an empty array with half the dimensions of the input image

'''Write your code here'''

print('Max-pooled Image Dimensions:',maxpooled.shape) # Prints dimensions of the maxpooled image

cv2.imwrite('maxpooled.png',maxpooled) # Stores image