Nama : Muhamad Arif Maulana

Nim : 1207070069

Kelas : TSEB

**Brightness**

Source code

import numpy as np #memanggil library numpy

import imageio #memanggil library imageio

import matplotlib.pyplot as plt #memanggil library matplotlib

img = imageio.imread("foto/R25.jpg") #memanggil foto pada folder

img\_height = img.shape[0] #untuk mengetahui tinggi gambar

img\_width = img.shape[1] #untuk mengetahui lebar gambar

img\_channel = img.shape[2] #untuk mengakses warna

img\_type = img.dtype #untuk membaca gambar

img\_brightness = np.zeros(img.shape, dtype=np.uint8) #mengosongkan aray untuk menyimpan nilai pixel

def brighter(nilai):

    for y in range(0, img\_height): #nilai y untuk tinggi

        for x in range(0, img\_width): #nilai x untuk lebar

            red = img[y][x][0]

            green = img[y][x][1]

            blue = img[y][x][2]

            gray = (int(red) + int(green) + int(blue)) / 3 #rumus mengkonvert ke grayscale

            gray += nilai #rumus konversi gray

            if gray > 255:

                gray = 255

            if gray < 0:

                gray = 0

            img\_brightness[y][x] = (gray, gray, gray) #mengeksekusi gambar dengan keluaran grayscale

brighter(-100) #perintah grayscale untuk menggelapkan gambar

plt.imshow(img\_brightness) #memanggil hasil gambar yang sudah di konversikan ke grayscale

plt.title("Brightness -100") #menamai hasil dari gambar

plt.show() #menampilkan hasil

brighter(100) #perintah grayscale untuk mencerahkan gambar

plt.imshow(img\_brightness) #memanggil gambar

plt.title("Brightness 100") #menamai hasil dari gambar

plt.show() #menampilkan hasil

img\_rgbbrightness = np.zeros(img.shape, dtype=np.uint8) #mengosongkan aray untuk menyimpan nilai pixel

def rgbbrighter(nilai):

    for y in range(0, img\_height): #nilai y untuk tinggi

        for x in range(0, img\_width): #nilai y untuk lebar

            red = img[y][x][0]

            red += nilai #rumus mengkonversi rgb

            if red > 255:

                red = 255

            if red < 0:

                red = 0

            green = img[y][x][1] #rumus mengkonversi rgb

            green += nilai

            if green > 255:

                green = 255

            if green < 0:

                green = 0

            blue = img[y][x][2] #rumus mengkonversi rgb

            blue += nilai

            if blue > 255:

                blue = 255

            if blue < 0:

                blue = 0

            img\_rgbbrightness[y][x] = (red, green, blue) #mengeksekusi gambar dengan keluaran RGB

rgbbrighter(-100) #perintah rgb untuk menggelapkan gambar

plt.imshow(img\_rgbbrightness) #memanggil hasil gambar yang sudah di konversikan ke rgb

plt.title("Brightness -100") #menamai gambar

plt.show() #menampilkan hasil gambar

rgbbrighter(100) #perintah rgb untuk menggelapkan gambar

plt.imshow(img\_rgbbrightness) #memanggil hasil gambar yang sudah di konversikan ke rgb

plt.title("Brightness 100") #menamai gambar

plt.show() #menampilkan hasil gambar

img\_contrass = np.zeros(img.shape, dtype=np.uint8) #mengosongkan aray untuk menyimpan nilai pixel

def contrass(nilai):

    for y in range(0, img\_height): #nilai y untuk tinggi

        for x in range(0, img\_width): #nilai x untuk lebar

            red = img[y][x][0]

            green = img[y][x][1]

            blue = img[y][x][2]

            gray = (int(red) + int(green) + int(blue)) / 3 #rumus untuk mengkonvert ke grayscale

            gray \*= nilai #rumus konversi grayscale

            if gray > 255:

                gray = 255

            img\_contrass[y][x] = (gray, gray, gray) #mengeksekusi gambar dengan keluaran RGB

contrass(2) #perintah kontras

plt.imshow(img\_contrass) #memanggil hasil gambar yang sudah di konversi

plt.title("Contrass 2") #menamai hasil gambar

plt.show() #menampilkan hasil gambar

contrass(3) #perintah kontras

plt.imshow(img\_contrass) #memanggil hasil gambar yang sudah di konversi

plt.title("Contrass 3") #menamai hasil gambar

plt.show() #menampilkan hasil gambar

img\_autocontrass = np.zeros(img.shape, dtype=np.uint8) #mengosongkan aray untuk menyimpan nilai pixel

def autocontrass():

    xmax = 300

    xmin = 0

    d = 0

    # Mendapatkan nilai d, dimana nilai d ini akan berpengaruh pada hitungan

    # untuk mendapatkan tingkat kontras

    for y in range(0, img\_height): #nilai y untuk tinggi

        for x in range(0, img\_width): #nilai x untuk lebar

            red = img[y][x][0]

            green = img[y][x][1]

            blue = img[y][x][2]

            gray = (int(red) + int(green) + int(blue)) / 3 #rumus untuk mengkonvert ke grayscale

            if gray < xmax:

                xmax = gray

            if gray > xmin:

                xmin = gray

    d = xmin-xmax #d adalah hasil dari nilai xmin-xmax

    for y in range(0, img\_height): #nilai y untuk tinggi

        for x in range(0, img\_width): #nilai x untuk lebar

            red = img[y][x][0]

            red = img[y][x][0]

            green = img[y][x][1]

            blue = img[y][x][2]

            gray = (int(red) + int(green) + int(blue)) / 3 #rumus untuk mengkonvert ke grayscale

            gray = int(float(255/d) \* (gray-xmax)) #rumus auto kontras

            img\_autocontrass[y][x] = (gray, gray, gray) #mengeksekusi gambar dengan keluaran grayscale

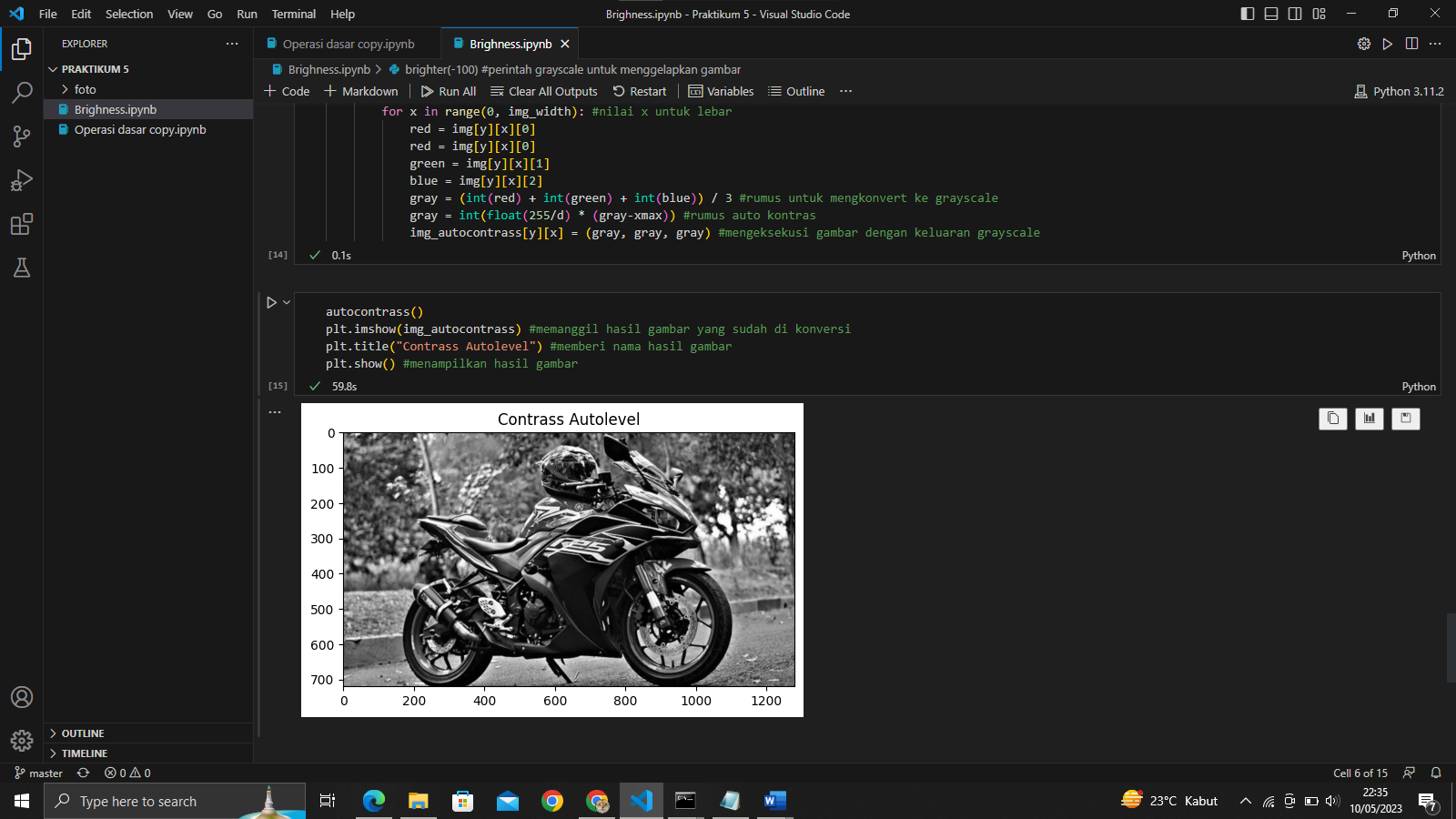
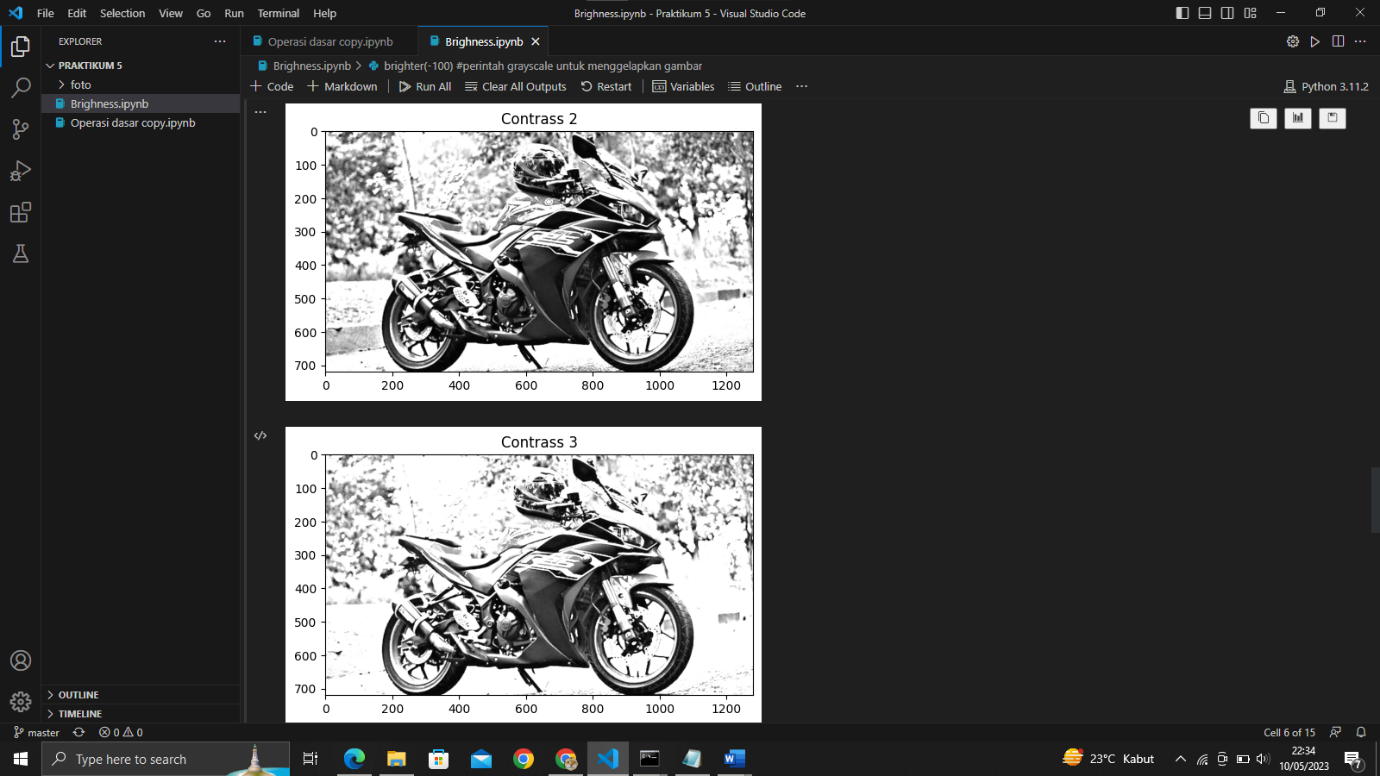
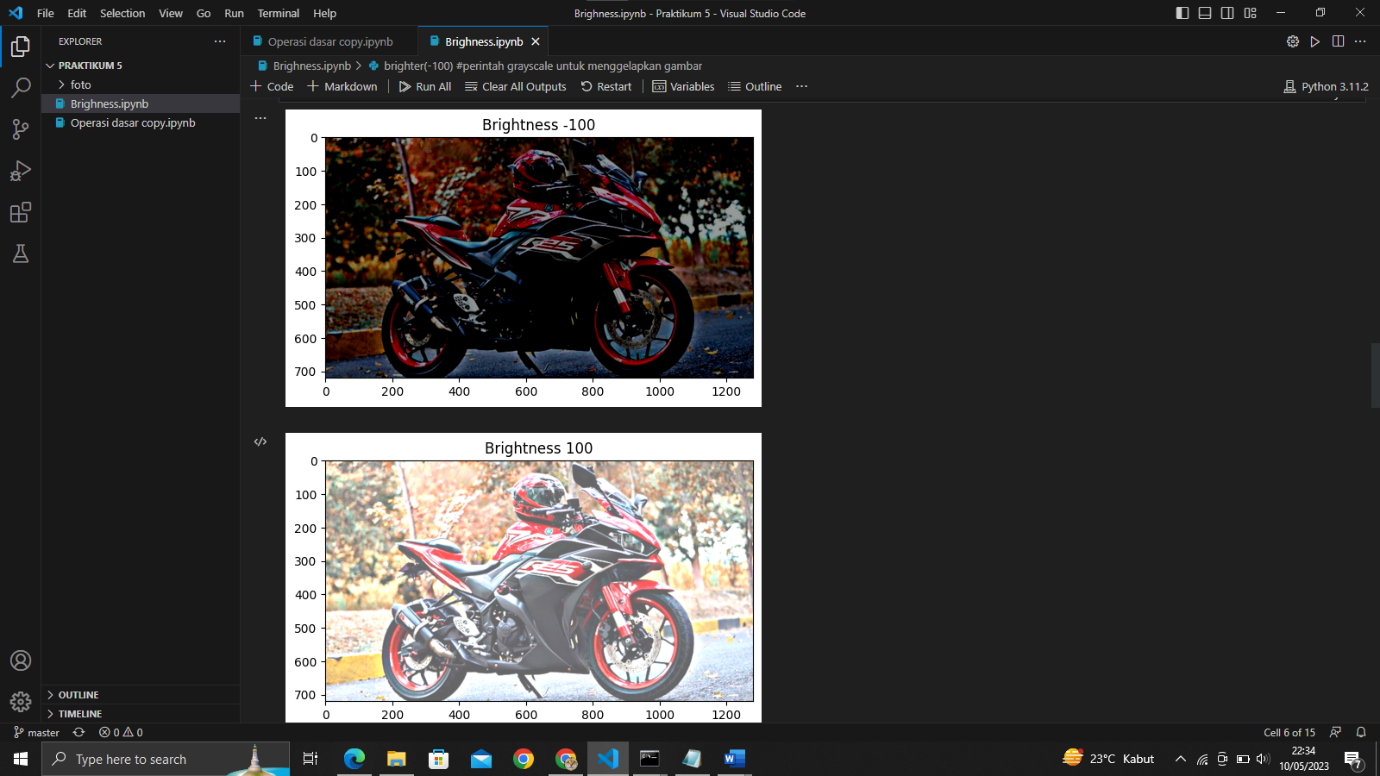
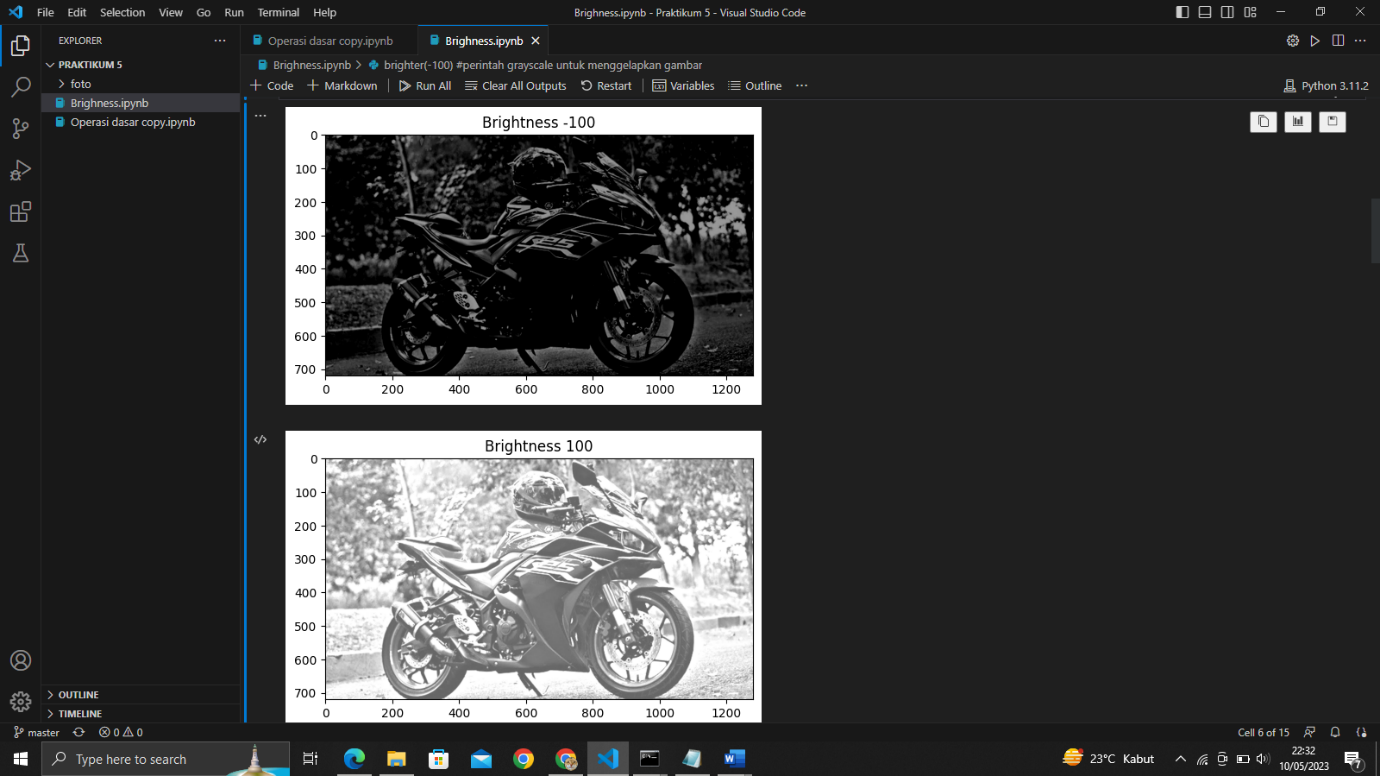
autocontrass()

plt.imshow(img\_autocontrass) #memanggil hasil gambar yang sudah di konversi

plt.title("Contrass Autolevel") #memberi nama hasil gambar

plt.show() #menampilkan hasil gambar

**Hasil tampilan**

****

**Operasi Dasar**

Source code

import matplotlib.pyplot as plt

%matplotlib inline

from skimage import data

from skimage import io

from skimage.io import imread

from skimage.color import rgb2gray

from skimage.util import invert

import numpy as np

foto1 = io.imread ("foto/R25.jpg")

foto2 = io.imread ("foto/mobil.jpg")

foto1\_Cropped = foto1.copy()

foto1\_Cropped = foto1\_Cropped [0:246,64:310]

foto2\_Cropped = foto2.copy()

foto2\_Cropped = foto2\_Cropped [64:246,128:310]

print('R25 Ori Shape : ',foto1.shape)

print('R25 Crop Shape : ',foto1\_Cropped.shape)

print('Mobil Ori Shape : ',foto2.shape)

print('Mobil Crop Shape : ',foto2\_Cropped.shape)

fig, axes = plt.subplots(2, 2, figsize=(12, 12))

ax = axes.ravel()

ax[0].imshow(foto1)

ax[0].set\_title("Citra Input 1")

ax[1].imshow(foto2, cmap='gray')

ax[1].set\_title('Citra Input 2')

ax[2].imshow(foto1\_Cropped)

ax[2].set\_title("Citra Output 1")

ax[3].imshow(foto2\_Cropped, cmap='gray')

ax[3].set\_title('Citra Output 2')

inv = invert(foto1\_Cropped)

print('Shape Input : ', foto1\_Cropped.shape)

print('Shape Output : ',inv.shape)

fig, axes = plt.subplots(2, 2, figsize=(12, 12))

ax = axes.ravel()

ax[0].imshow(foto1\_Cropped)

ax[0].set\_title("Citra Input")

ax[1].hist(foto1\_Cropped.ravel(), bins=256)

ax[1].set\_title('Histogram Input')

ax[2].imshow(inv)

ax[2].set\_title('Citra Output (Inverted Image)')

ax[3].hist(inv.ravel(), bins=256)

ax[3].set\_title('Histogram Output')

copyFoto = foto2\_Cropped.copy().astype(float)

shape = copyFoto.shape

output1 = np.empty(shape)

for baris in range(0, shape[0]):

    for kolom in range(0, shape[1]):

        a1 = baris

        b1 = kolom

        output1[a1, b1] = copyFoto[baris, kolom] /100

fig, axes = plt.subplots(2, 2, figsize=(12, 12))

ax = axes.ravel()

ax[0].imshow(foto2\_Cropped, cmap='gray')

ax[0].set\_title("Citra Input")

ax[1].hist(foto1\_Cropped.ravel(), bins=256)

ax[1].set\_title('Histogram Input')

ax[2].imshow(output1, cmap='gray')

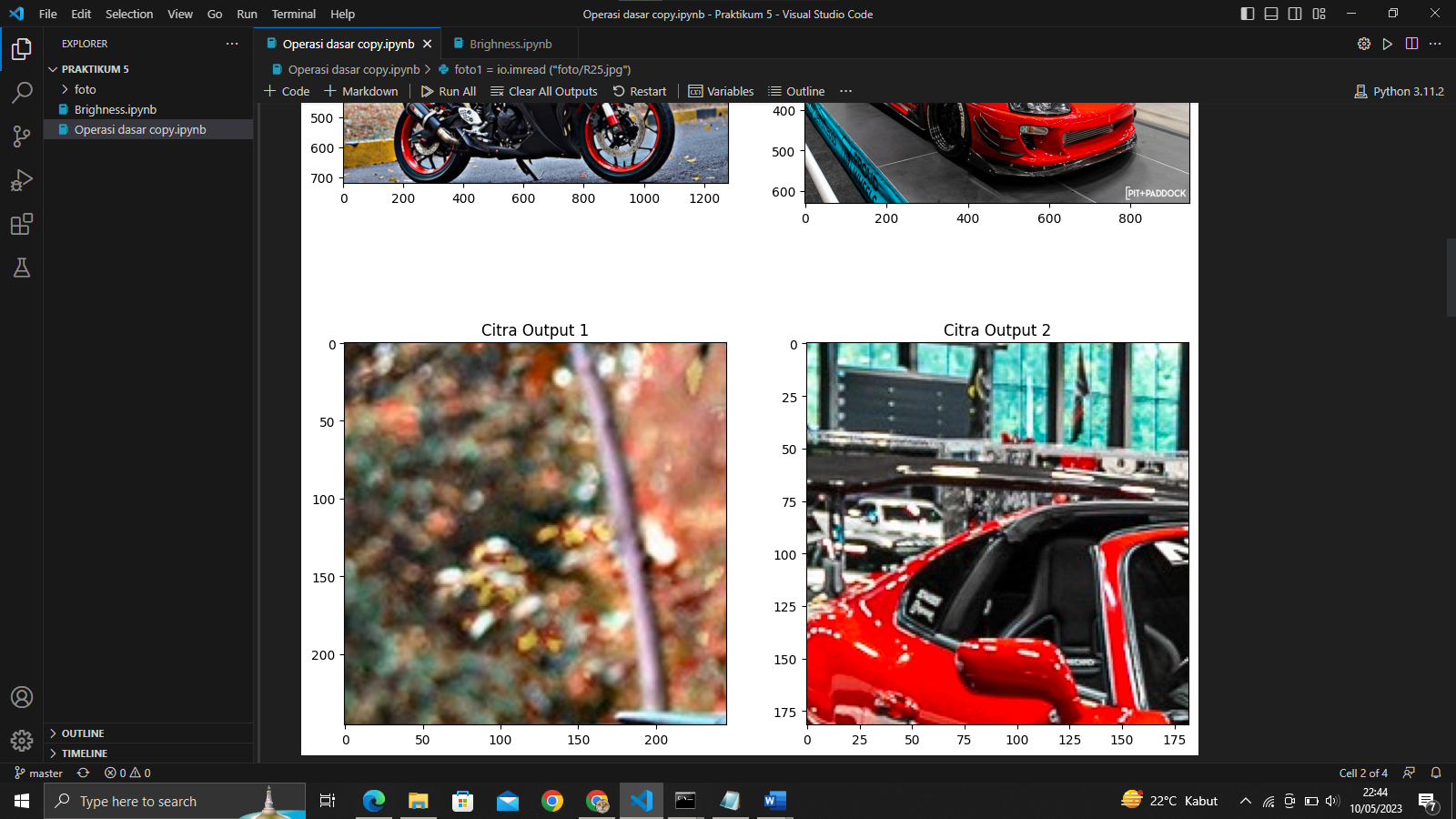
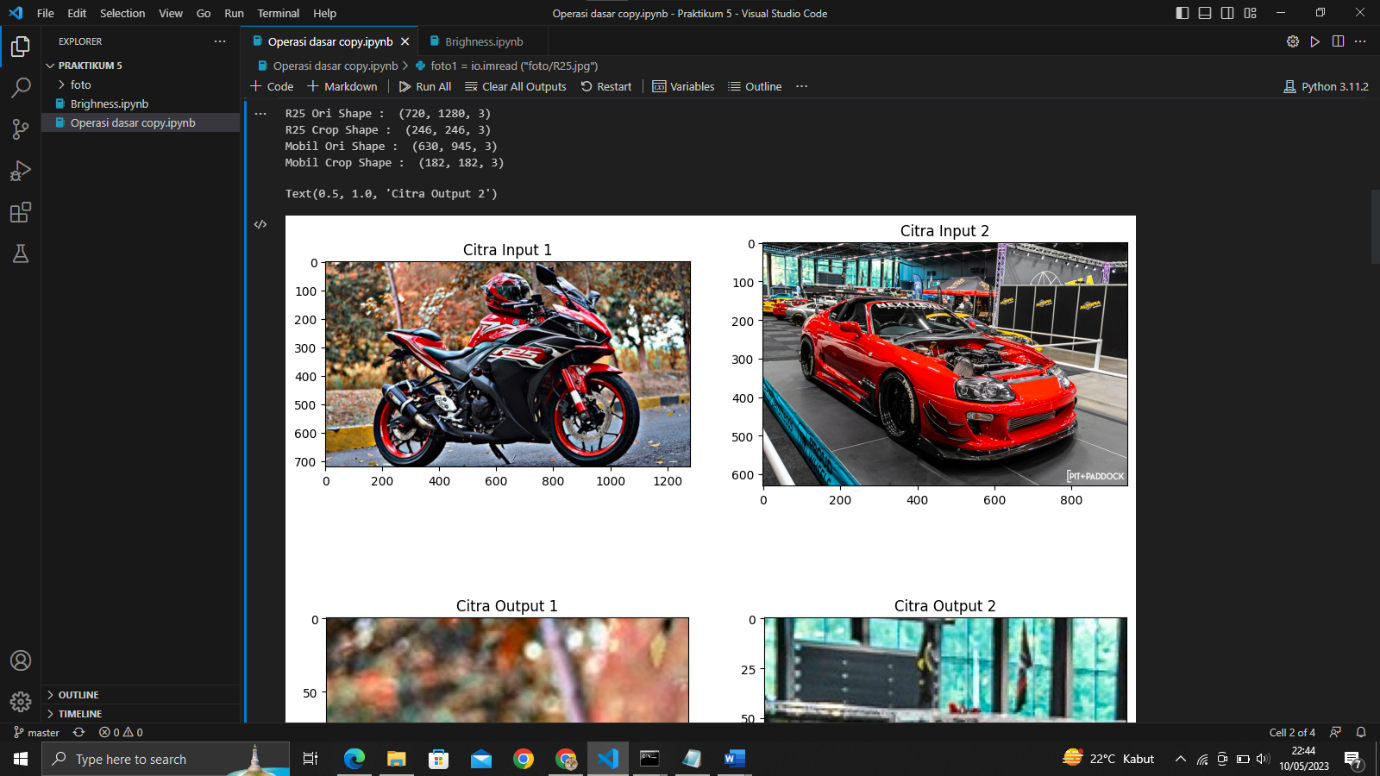
ax[2].set\_title('Citra Output (Brightnes)')

ax[3].hist(output1.ravel(), bins=256)

ax[3].set\_title('Histogram Input')

Hasil tampilan

Cropping image



Citra Negative

