ĐẠI HỌC QUỐC GIA TP.HCM

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

CHƯƠNG TRÌNH CHẤT LƯỢNG CAO

LỚP: 20CLC08

ĐỒ ÁN 2 – IMAGE PROCESSING

MÔN: TOÁN ỨNG DỤNG – THỐNG KÊ

HỌ VÀ TÊN: LÊ ĐỨC ĐẠT

MSSV: 20127674

TP.HCM, 5/7/2022

MỤC LỤC

I. Các chức năng đã hoàn thành:

STT	CÁC CHỨC NĂNG	TIẾN ĐỘ
1	Thay đổi độ sáng cho ảnh	100%
2	Thay đổi độ tương phản	100%
3	Ảnh màu -> Ảnh xám	100%
4	Xoay ảnh ngang-dọc	100%
5	Chồng 2 ảnh cùng kích thước	100%
6	Làm mờ ảnh	100%
7	Hàm main	100%

II. Ý tưởng thực hiện, mô tả các hàm, các chức năng…:

Ảnh gốc (1200x800):

- Khai báo thư viện: Khai báo thư viện phù hợp với yêu cầu đề bài

```
In [17]: import numpy as np
         from PIL import Image
         import matplotlib.pyplot as plt
```

- Đọc ảnh: Ta đọc 1 ảnh dựa vào hướng dẫn của ảnh muốn đọc (ta gọi đó là img_path) và trả về 1 mảng numpy các điểm ảnh của ảnh đó.

```
In [18]: # Read Image
         def imageReading(img_path):
             return np.array(Image.open(img_path))
```

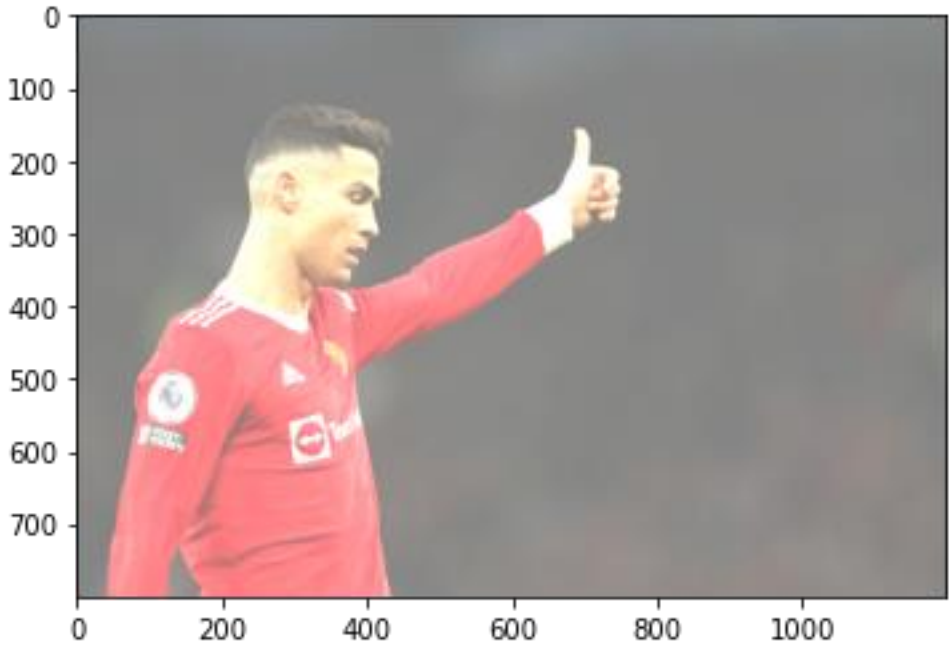
1. Thay đổi độ sáng của ảnh:

- Input: Đường dẫn của ảnh, kèm theo 1 tham số thay đổi độ sáng(n).

  Output: Ảnh kiểu Image **(ghét của nào trời trao của nó, .jpg->.jpg)**. Độ sáng tỉ lệ thuận với n (lưu ý: n tiến tới vô hạn thì ảnh có thể trắng xóa hoặc tối mịt). Từ đó, lên ý tưởng: Cộng n vào kênh màu, max = 255, nếu vượt quá thì lấy max.

```
In [19]: #Câu 1: Thay đổi độ sáng cho ảnh (1 điểm)
         def cau1(img_path, n):
             img = imageReading(img_path) + float(n)
             img = np.clip(img, 0, 255)
             return Image.fromarray(img.astype(np.uint8))
```

➔ Kết quả (với n = 120):



2. Thay đổi độ tương phản của ảnh:

Tương tự như tắng sáng cho ảnh ở câu 1, chỉ khác:

- Giá trị n tỉ lệ thuận với độ tương phản.
- Ta sẽ thay đổi tất cả kênh màu của từng ô trong mảng numpy theo công thức:

  newmodel = oldModel x F – 128 * F + 128

  Trong đó:

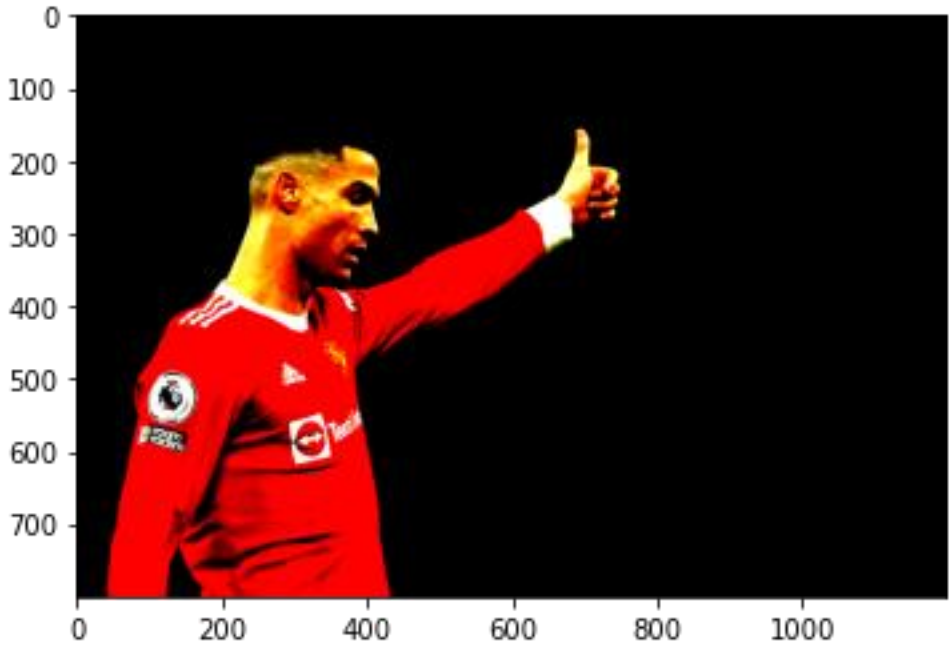
  + newModel: Giá trị sau khi áp dụng oldModel

  + F = 259 x (255 + n) / (255 x (259 – n)).

  Nếu giá trị kênh nào vượt quá 255 thì lấy max = 255.

```
In [20]: #Câu 2: Thay đổi độ tương phản (1 điểm)
def cau2(img_path, n):
    f = (259 * (255 + n)) / (255 * (259 - n))
    img = imageReading(img_path) * float(f) - 128 * f + 128
    img = np.clip(img, 0, 255)
    return Image.fromarray(img.astype(np.uint8))
```

➔ Kết quả (n = 150)



3. Ảnh màu->Ảnh xám:

   Input: Ảnh màu

   Output: ảnh xám

   Ý tưởng: Với từng pixel trong mảng numpy được tạo ở trên, ta sẽ gộp 3 kênh màu thành 1 giá trị constant [0, 255] để biến ảnh RGB thành ảnh xám. Mỗi kênh màu sẽ có 1 trọng số tương ứng là 0.3, 0.59 và 0.11 ứng với 3 kênh màu lần lượt là Đỏ (Red), Xanh lá cây (Green) và Xanh dương (Blue).

   Giá trọ mỗi pixel trong mảng numpy sẽ tính:

   grayscalePixel = Red x 0.3 + Green x 0.59 + Blue x 0.11

Ta tận dụng sức mạnh của thư viện Numpy để nhân vector trọng số với ma trận của ảnh.

```
In [21]: #Câu 3: Chuyển đổi ảnh màu thành ảnh xám (2 điểm)
         def cau3(img_path):
             img = imageReading(img_path) @ np.array([0.3, 0.59, 0.11])
             return Image.fromarray(img.astype(np.uint8))
```

(Ảnh không xám lắm mong các Thầy Cô châm chước cho ạ hehe).

4. Lật ảnh:

a. Ngang:

Input: như các câu trên

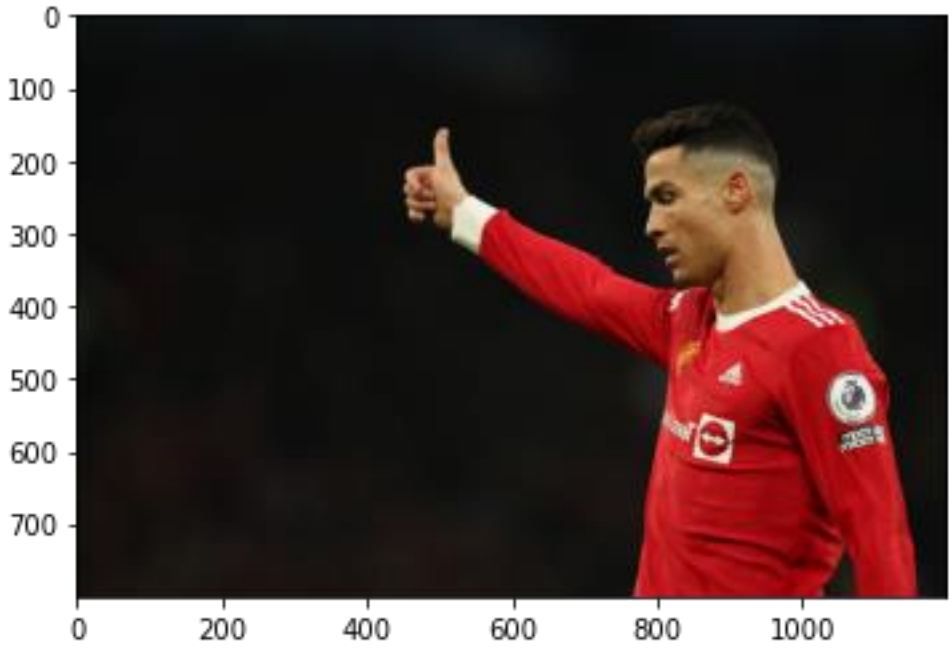
Output: Ảnh được lật ngang

Ý tưởng: Hoán đổi vị trí các cột của bức ảnh, cũng giống như hoán đổi các cột ma trận, hay nói chính xác hơn là hoán đổi các số trong chuỗi tang dần thành giảm dần rồi in kết quả.

Dùng hàm flip của thư viện Numpy để có thể thục hiện được.

```
In [22]: #Câu 4.1: Xoay ảnh ngang (1 điểm)
         def cau41(img_path):
             return Image.fromarray(np.flip(imageReading(img_path), axis=1))
```

➔ Kết quả:



b. Dọc:

Input: Như các câu trên

Output: Ảnh đã được lật dọc

Ý tưởng: Như xoay ngang, chỉ khác ở chỗ: Ta xoay các dòng của từng pixel trong ảnh - > xoay các dòng trong ma trận.

5. Chồng 2 ảnh cùng kích thước:

Input: 2 ảnh cùng kích thước 1200x800

Output: 1 ảnh sau khi được chồng (dĩ nhiên là ảnh xám).

Ý tưởng: Đổi 2 ảnh màu thành 2 ảnh xám bằng cách gọi hàm câu 3, rồi cộng 2 ảnh với trọng số tượng ứng dưới công thức:

Pixel = pixel_of_pic1 x w1 + pixel_of_pic2 x w2

Với w1, w2 lần lượt là trọng số của ảnh 1 và ảnh 2, w1, w2 thuộc [0,1], w1 = 1 – w2. Nếu muốn ảnh rõ hơn thì cho trọng số của ảnh đó cao hơn, giá trị mặc định của w1 là 0.5->trọng số, độ rõ của 2 ảnh là như nhau.

```
In [24]: #Câu 5: Chồng 2 ảnh cùng kích thước 1200 x 800 (Only ảnh xám) (1 điểm)
def cau5(img_path_1, img_path_2, weight_1 = 0.5):
    weight_2 = 1 - weight_1
    img = np.array(cau3(img_path_1)) * weight_1\
        + np.array(cau3(img_path_2)) * weight_2
    return Image.fromarray(img.astype(np.uint8))
```

➔ Kết quả:

6. Làm mờ ảnh:

   Input: Ảnh nào đó

   Output: Ảnh được làm mờ

   Ý tưởng: dung ma trận kernel kiểu Gaussian blur 3x3 làm ma trận trọng số. Sau đó, duyệt từng pixel là làm mờ chúng bằng cách lấy trung các giá trị pixel xung quanh pixel đang xét ứng với trọng số tương ứng trong ma trận kernel. Ta đã them padding các pixel [0,0,0] vào ma trận ảnh trước. Sau khi làm mờ tất cả các pixel, ta bỏ các padding đi và trả về nội dung ảnh, cũng là kết quả.

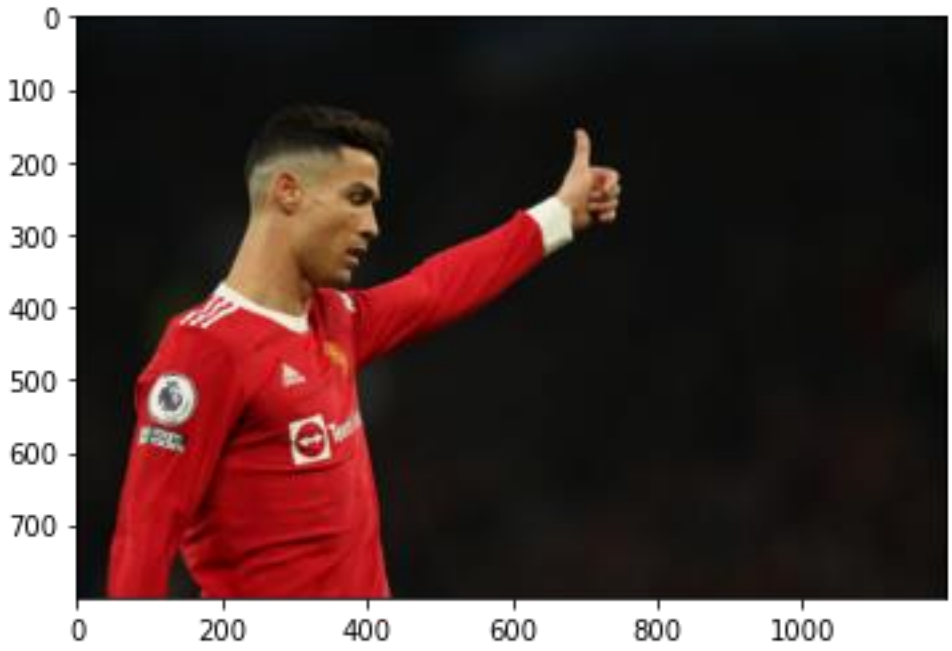
In [25]:
```
#Câu 6: Làm mờ ảnh (2 điểm)
def cau6(img_path):
    kernel = np.array([[[1], [2], [1]],
                       [[2], [4], [2]],
                       [[1], [2], [1]]]) / 16

    img = imageReading(img_path)
    temp = np.zeros((img.shape[0] + 2, img.shape[1] + 2, img.shape[2]))
    temp[1:-1,1:-1,:] = img

    for col in range(0, img.shape[0]):
        for row in range(0, img.shape[1]):
            img[col][row] = (temp[col:col+3, row:row+3] * kernel).sum(axis=1).sum(axis=0)

    return Image.fromarray(img.astype(np.uint8))
```

➔ Kết quả:



Không mờ lắm nhưng tạm chấp nhận được.

7. Hàm main:

   Tạo như cách tạo menu trong C++.

   Có thể cho phép người du.ng chọn chức năng.

In [31]:
```
#7. Hàm main (1 điểm)
if __name__ == "__main__":
    file1 = input("Nhập đường dẫn file 1: ")
    file2 = input("Nhập đường dẫn file 2: ")
    choice = input("Sự lựa chọn của bạn là (1, 2, 3, 41(ngang), 42(doc), 5, 6, 0): ")
    if choice == '1':
        anh1 = cau1(file1, 120)
        plt.imshow(anh1)
    if choice == '2':
        anh2 = cau2(file1, 150)
        plt.imshow(anh2)
    if choice == '3':
        anh3 = cau3(file1)
        plt.imshow(anh3)
    if choice == '41':
        anh41 = cau41(file1)
        plt.imshow(anh41)
    if choice == '42':
        anh42 = cau42(file1)
        plt.imshow(anh42)
    if choice == '5':
        anh5 = cau5(file1, file2, 0.5)
        plt.imshow(anh5)
    if choice == '6':
        anh6 = cau6(file1)
        plt.imshow(anh6)
    if choice == '0':
        anh1 = cau1(file1, 120)
        plt.imshow(anh1)
        plt.savefig('anh1.jpg')
        anh2 = cau2(file1, 150)
        plt.imshow(anh2)
        plt.savefig('anh2.jpg')
        anh3 = cau3(file1)
        plt.imshow(anh3)
        plt.savefig('anh3.jpg')
        anh41 = cau41(file1)
        plt.imshow(anh41)
        plt.savefig('anh41.jpg')
        anh42 = cau42(file1)
        plt.imshow(anh42)
        plt.savefig('anh42.jpg')
        anh5 = cau5(file1, file2, 0.5)
        plt.imshow(anh5)
        plt.savefig('anh5.jpg')
        anh6 = cau6(file1)
        plt.imshow(anh6)
        plt.savefig('anh6.jpg')
```

```
Nhập đường dẫn file 1: C:\Users\ASUS\Desktop\a.jpg
Nhập đường dẫn file 2: C:\Users\ASUS\Desktop\b.jpg

Sự lựa chọn của bạn là (1, 2, 3, 41(ngang), 42(doc), 5, 6, 0):
```

Sẽ ra kết quả như các câu trên.

Riêng lựa chọn số 0, thì chỉ show 1 kết quả trên jupyter, sau đó tất cả các ảnh kết quả sẽ được lưu ở chính thư mục thao tác trên Jupyter Notebook bằng cách dung` hàm savefig trong thư viện matplotlib.

III. Tài liệu tham khảo:

- Github anh Kiều Công Hậu K18: https://github.com/kieuconghau/image-processing.
- Github anh Đoàn Đình Toàn K19: https://github.com/t3bol90/ST-MA-Lab03.
- https://www.geeksforgeeks.org/image-processing-without-opencv-python/.
- https://www.autoscripts.net/read-image-in-python-without-opencv/.