**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN 1**

**XỬ LÝ ẢNH**

*Người thực hiện*: **Nguyễn Phú Sang – 21023391**

**Lâm Phát Đạt – 22676051**

**Đồng Văn Xướng - 22679121**

Lớp **: 420300150101**

*Người hướng dẫn*: **ThS. Võ Quang Hoàng Khang**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024**

**NỘI DUNG BÁO CÁO**

**1. Thông tin sinh viên**

|  |  |
| --- | --- |
| **Mssv** | **Họ Tên** |
| 21023391 | Nguyễn Phú Sang |
| 22676051 | Lâm Phát Đạt |
| 22679121 | Đồng Văn Xướng |

**2. Thống kê mức độ hoàn thành**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Các chức năng** | **Mức độ hoàn thành** | **Sinh viên thực hiện** |
| 1 | Checkered Board | 100% | Lâm Phát Đạt |
| 2 | Color Correction | 100% | Lâm Phát Đạt |
| 3 | Rotate Image | 100% | Lâm Phát Đạt |
| 4 | Color Separation | 100% | Đồng Văn Xướng |
| 5 | Corner Line | 100% | Đồng Văn Xướng |
| 6 | Gradient | 100% | Đồng Văn Xướng |
| 7 | Letter B | 100% | Nguyễn Phú Sang |
| 8 | Find Sercet By Subtract | 100% | Nguyễn Phú Sang |

**3. Phân tích và mô tả thuật toán**

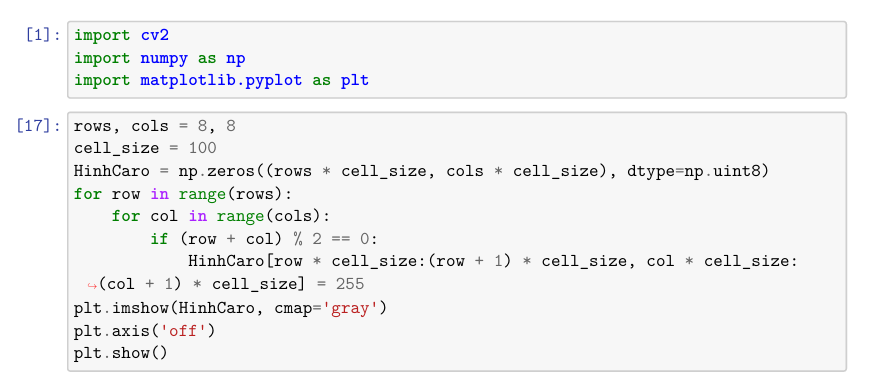
1. **Checkedred Board**

**- Ý tưởng:** Tạo ra một hình ảnh có dạng bàn cờ với các ô vuông đen và trắng xen kẽ.Cách làm là tạo một hình ảnh trống, sau đó duyệt qua từng vị trí trên bàn cờ và tô màu các ô dựa trên vị trí của chúng. Nếu vị trí của ô nằm trên hàng và cột có tổng là số chẵn, ô đó được tô màu trắng, nếu là số lẻ thì giữ nguyên màu đen.

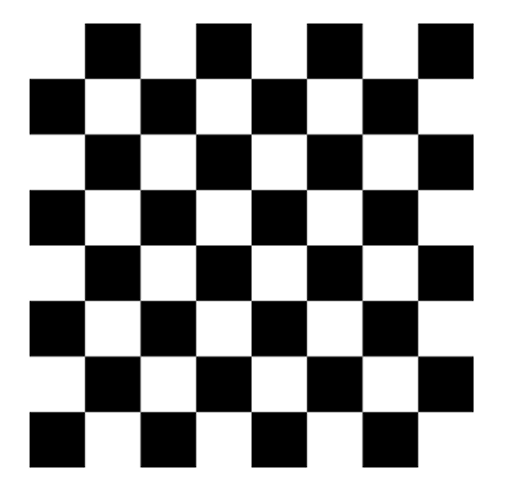
**- Mô tả thuật toán:**

* Khởi tạo tham số: Đặt số hàng và số cột là 8, và kích thước ô là 100.
* Tạo ảnh trống: Sử dụng np.zeros để tạo một lưới 8x8 với mỗi ô có kích thước 100x100 pixel.
* Điền vào lưới: Lặp qua từng ô trong lưới. Nếu tổng của chỉ số hàng và cột là số chẵn, điền ô đó bằng màu trắng (255). Ngược lại, để ô đó màu đen (0).
* Hiển thị ảnh: Sử dụng plt.imshow để hiển thị ảnh dưới dạng thang độ xám và tắt trục để ảnh trông gọn gàng hơn.

**- Code:**

****

**- Kết quả:**

****

1. **Color Corection**

**- Ý tưởng:** Thực hiện chuyển đổi một hình ảnh màu thành hình ảnh xám, sau đó tạo ra và hiển thị ảnh xám.

**- Mô tả thuật toán:**

* Đọc ảnh: Sử dụng cv2.imread để đọc ảnh từ đường dẫn đã chỉ định.
* Chuyển đổi sang ảnh xám: Tạo ảnh xám bằng cách kết hợp các kênh màu đỏ, xanh lá và xanh dương với các trọng số tương ứng (0.07, 0.72, 0.21).
* Hiển thị ảnh: Sử dụng plt.imshow để hiển thị ảnh xám và tắt trục để ảnh trông gọn gàng hơn.

**- Code:**



**- Kết quả:**



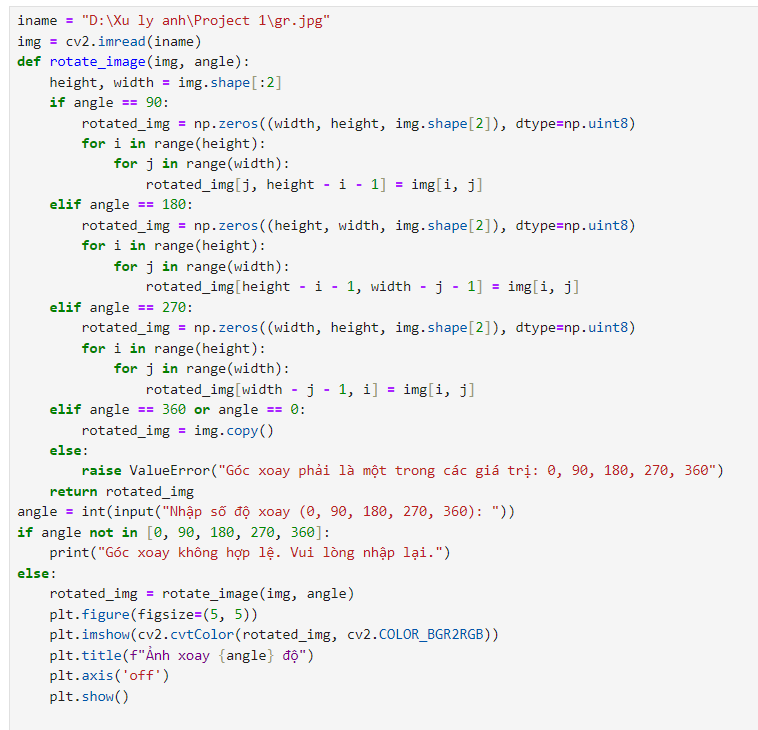
1. **Rotate Image**

**- Ý tưởng:** Xoay một hình ảnh theo các góc cố định (0, 90, 180, 270, 360 độ) và hiển thị kết quả.

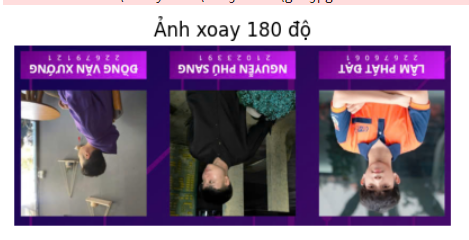
**- Mô tả thuật toán:**

* Chuyển đổi góc độ sang radian: Góc độ mà người dùng nhập vào được chuyển đổi từ đơn vị độ sang radian để phục vụ cho các phép toán lượng giác.
* Tính toán kích thước mới cho ảnh: Khi xoay ảnh, kích thước của ảnh có thể thay đổi. Thuật toán tính toán chiều rộng và chiều cao mới của ảnh dựa trên góc độ xoay.
* Tạo một ảnh rỗng mới: Một mảng ảnh mới với kích thước đã tính toán được tạo ra, tất cả các pixel ban đầu đều được gán giá trị 0.
* Xác định tâm của ảnh cũ và mới: Xác định tọa độ tâm của ảnh gốc và ảnh mới để làm cơ sở cho các phép tính dịch chuyển.
* Duyệt qua từng pixel trong ảnh mới: Với mỗi pixel trong ảnh mới, thuật toán tính toán vị trí tương ứng của nó trong ảnh gốc dựa trên góc xoay và vị trí của tâm ảnh.
* Kiểm tra và gán giá trị cho pixel: Nếu pixel trong ảnh gốc nằm trong phạm vi hợp lệ, giá trị của nó sẽ được gán cho pixel tương ứng trong ảnh mới.
* Hiển thị ảnh đã xoay: Ảnh đã xoay được hiển thị bằng cách sử dụng thư viện Matplotlib.

**- Code:**



**- Kết quả:**



1. **Color Separation**

**- Ý tưởng:** Phân tách đối tượng trong ảnh đen trắng.

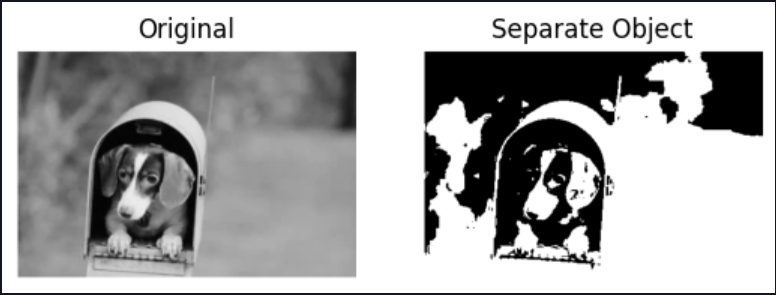
**- Mô tả thuật toán:**

* Đọc ảnh: Đọc ảnh đầu vào dưới dạng ảnh xám (grayscale) với OpenCV.
* Đặt ngưỡng: Xác định một giá trị ngưỡng (threshold), ví dụ là 128.
* Khởi tạo ảnh nhị phân: Tạo một mảng mới binary\_image với cùng kích thước như ảnh gốc, ban đầu tất cả các giá trị pixel được đặt là 0 (đen).
* Phân loại các pixel: Lặp qua từng pixel của ảnh gốc. Sau đó sánh giá trị của mỗi pixel với giá trị ngưỡng: Nếu giá trị pixel lớn hơn ngưỡng, pixel tương ứng trong ảnh nhị phân được đặt là 255 (trắng). Nếu giá trị pixel nhỏ hơn hoặc bằng ngưỡng, pixel tương ứng được đặt là 0 (đen).
* Hiển thị kết quả cạnh nhau để dễ so sánh.

· **- Code:**



**- Kết quả:**



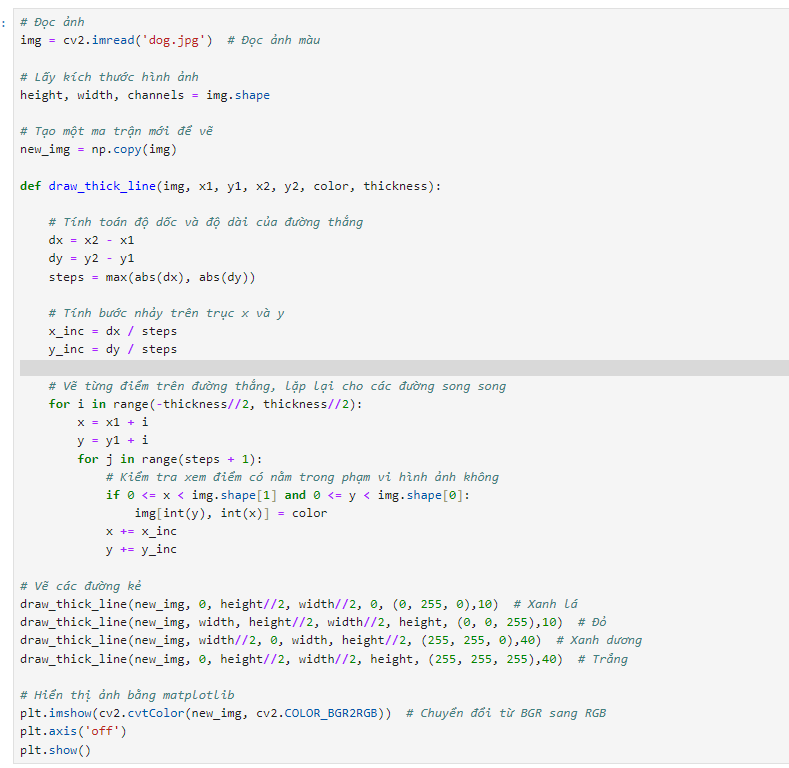
1. **Corner Line**

**- Ý tưởng:** Tạo các đường line ở góc của bức ảnh.

**- Mô tả thuật toán:**

* Đọc ảnh đầu vào: Sử dụng cv2.imread để đọc ảnh.
* Hàm resize\_with\_aspect\_ratio: Thay đổi kích thước ảnh mà vẫn giữ nguyên tỷ lệ khung hình.
* Hàm draw\_thick\_line: Vẽ các đường thẳng với độ dày xác định trên ảnh.
* Vòng lặp chính: Lặp qua các tọa độ xác định (x, y, width, height) và áp dụng các hàm như vẽ hình chữ nhật (cv2.rectangle) và văn bản (cv2.putText) lên ảnh đầu vào.
* Xuất ảnh: Ảnh đầu ra sẽ được chú thích với các hình chữ nhật và văn bản tại các vị trí xác định**.**

**- Code:**



**- Kết quả:**



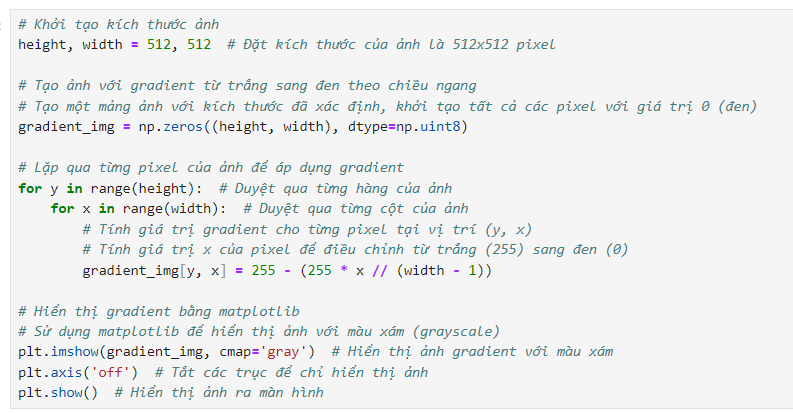
1. **Gradient**

**- Ý tưởng:** Tạo ra một ảnh gradient từ trắng sang đen theo chiều ngang và hiển thị nó.

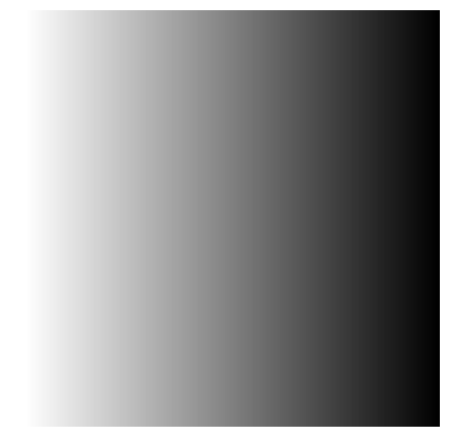
**- Mô tả thuật toán:**

* Khởi tạo kích thước ảnh: Đặt kích thước của ảnh là 512x512 pixel.
* Tạo mảng ảnh: Tạo một mảng 2 chiều với kích thước 512x512, khởi tạo tất cả các giá trị pixel bằng 0 (màu đen).
* Áp dụng gradient: Lặp qua từng hàng (y) và từng cột (x) của ảnh. Tính giá trị của mỗi pixel dựa trên vị trí của nó theo chiều ngang: Giá trị của pixel tại vị trí (y, x) được tính bằng công thức 255 - (255 \* x // (width - 1)), nơi x là chỉ số cột và width - 1 là giá trị tối đa của cột. Các giá trị pixel giảm từ 255 (trắng) ở cột đầu tiên đến 0 (đen) ở cột cuối cùng.

**- Code:**



**- Kết quả:**



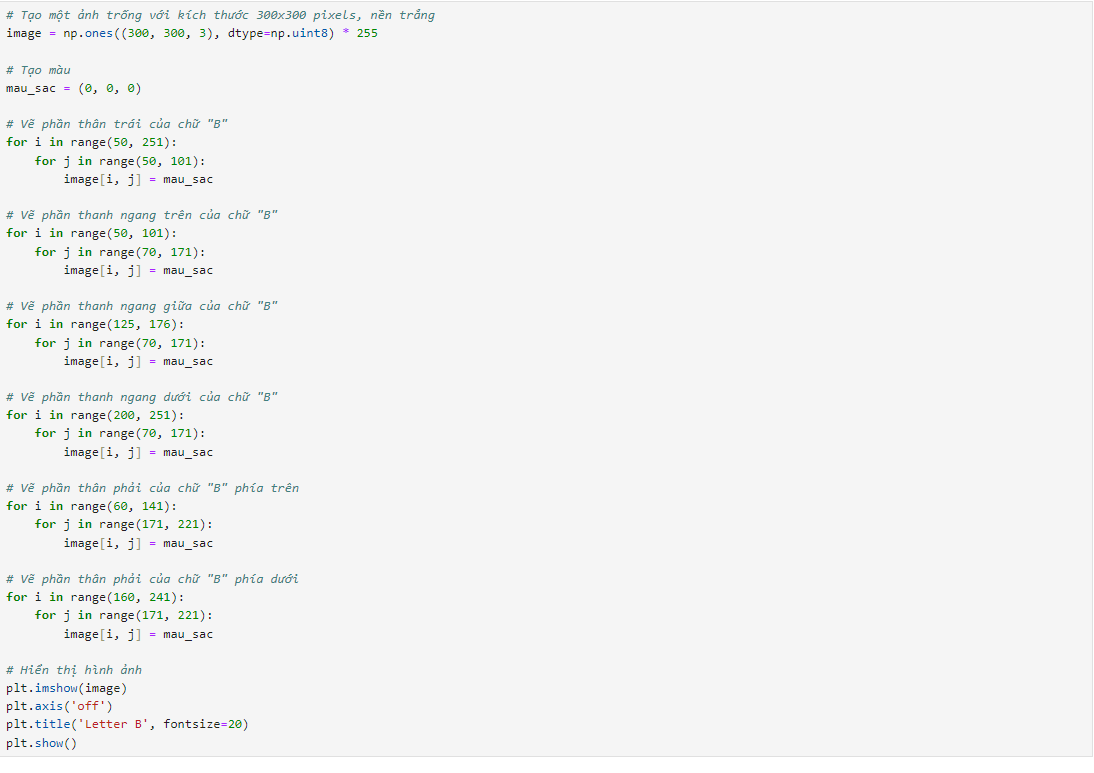
1. **Letter B**

**- Ý tưởng:** Thiết kế chữ B bằng numpy và open-cv.

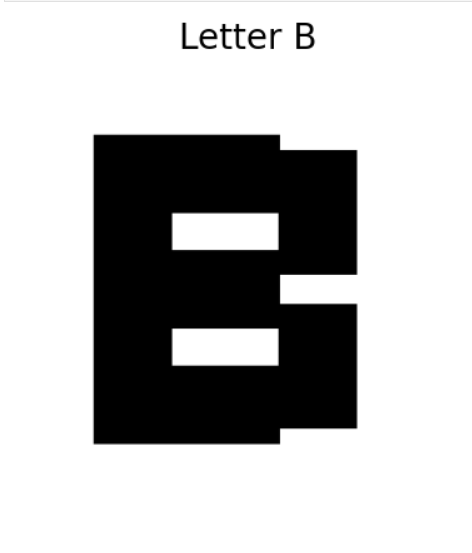
**- Mô tả thuật toán:**

* Khởi tạo ảnh nền: Tạo một ảnh trống với kích thước 300x300 pixels, nền màu trắng.
* Chọn màu sắc: Chọn màu đen cho các phần của chữ "B".
* Vẽ các phần của chữ "B":
* Thân trái: Vẽ một hình chữ nhật lớn cho phần thân trái của chữ "B".
* Các thanh ngang: Vẽ ba hình chữ nhật cho các thanh ngang trên, giữa, và dưới của chữ "B".
* Thân phải: Vẽ 2 hình chữ nhật có phần thân phải của chữ "B"
* Hiển thị ảnh: Sử dụng matplotlib để hiển thị hình ảnh với chữ "B".

**- Code:**



**- Kết quả:**

****

1. **Find Sercet By Subtract**

**- Ý tưởng:** Tìm bí mật bằng cách theo dõi 2 hình ảnh.

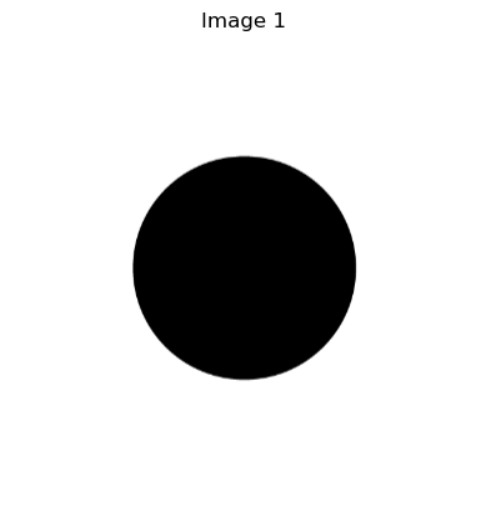
**- Mô tả thuật toán:**

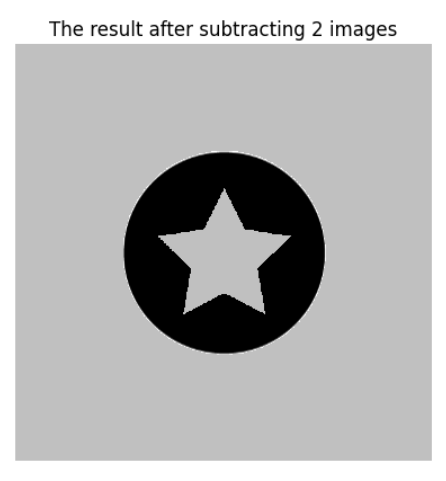
* Thêm hình ảnh: Tải hình ảnh từ file sau đó hiển thị ảnh gốc.
* Kiểm tra kích thước: Nếu kích thước 2 ảnh không cùng nhau sẽ không thể thực hiện phép toán.
* Khởi tạo một hình ảnh với nền trắng: Tạo một hình ảnh mới với nền trắng có cùng kích thước với 2 hình ảnh ban đầu.
* Thực hiện phép toán: Duyệt ảnh theo chiều cao và chiều rộng sau đó so sánh từng điểm ảnh của hai hình ảnh. Nếu điểm ảnh từ hai hình ảnh khác nhau, giữ lại điểm ảnh từ hình ảnh đầu tiên trong hình ảnh kết quả. Nếu giống nhau, thay bằng màu trắng.
* Hiển thị kết quả: Hiển thị hình ảnh kết quả sau phép toán.

**- Code:**



**- Kết quả:**

****



**4. Tài liệu tham khảo**

1. GeeksforGeeks. (2023). Làm cách nào để trừ hai hình ảnh bằng Python-OpenCV?, từ: https://www.geeksforgeeks.org/how-to-subtract-two-images-using-python-opencv/