1. Program to read and display given image

```
import cv2

# Load the image
image = cv2.imread(r"C:\Users\gioes\Desktop\yellow.jpg")  # Replace 'image_path.jpg' with the path to your image file

# Check if the image was successfully loaded
if image is None:
    print("Error: Image not found.")
else:
    # Display the image
    cv2.imshow('Displayed Image', image)

    # Wait indefinitely until a key is pressed
    cv2.waitKey(0)

    # Close all OpenCV windows
    cv2.destroyAllWindows()
```

input:

Output:

2. Program to display grayscale image of given image

```
import cv2
```

```
# Load the image (replace 'image_path.jpg' with the path to
your image file)
image = cv2.imread(r'C:\Users\gioes\Desktop\yellow.jpg')

# Check if the image was successfully loaded
if image is None:
    print("Error: Image not found.")
else:
    # Convert the image to grayscale
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # Display the grayscale image
    cv2.imshow('Grayscale Image', gray_image)

    # Wait indefinitely until a key is pressed
    cv2.waitKey(0)

    # Close all OpenCV windows
    cv2.destroyAllWindows()
```

input:

Output:

3. Program to display binary image of given image

```
import cv2

# Load the image (replace 'image_path.jpg' with the path to your image file)
image = cv2.imread(r'C:\Users\gioes\Desktop\yellow.jpg')

# Check if the image was successfully loaded
if image is None:
    print("Error: Image not found.")
else:
    # Convert the image to grayscale
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # Apply binary thresholding
    # The second argument is the threshold value, and the third is the max value to use with the THRESH_BINARY option
    _, binary_image = cv2.threshold(gray_image, 127, 255, cv2.THRESH_BINARY)

    # Display the binary image
    cv2.imshow('Binary Image', binary_image)

    # Wait indefinitely until a key is pressed
    cv2.waitKey(0)

    # Close all OpenCV windows
    cv2.destroyAllWindows()
```

input:

output:

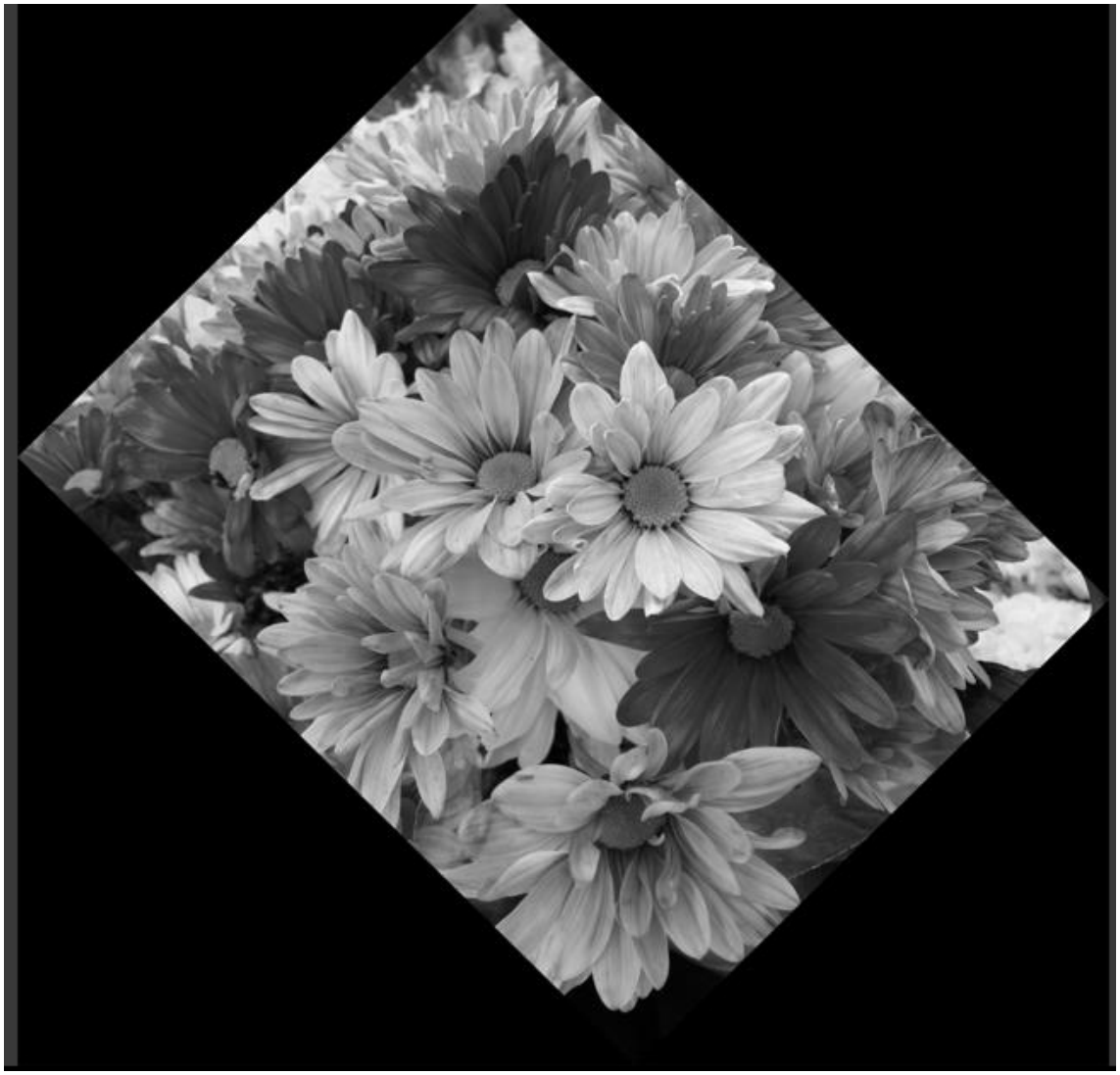
4.program to read grayscale image and rotate it

from PIL import Image

```
# Load the grayscale image
img = Image.open(r'C:\Users\Admin\Desktop\bunch.jpg').convert('L')
# 'L' mode for grayscale

# Rotate the image by 45 degrees
rotated_img = img.rotate(45, expand=True)

# Save the rotated image
rotated_img.save('rotated_image.jpg')

# Show the original and rotated images
img.show()
rotated_img.show()
```

Input:

Output:



5.program that reads a grayscale image and change the position of the image

```
import cv2

import numpy as np


# Load the grayscale image
```

```
image = cv2.imread(r'C:\Users\Admin\Desktop\bunch.jpg', cv2.IMREAD_GRAYSCALE)

# Define shift values
x_shift = 100  # Shift right
y_shift = 70  # Shift down

# Create the transformation matrix for shifting
M = np.float32([[1, 0, x_shift], [0, 1, y_shift]])

# Shift the image
shifted_image = cv2.warpAffine(image, M, (image.shape[1], image.shape[0]))

# Display the original and shifted images
cv2.imshow('Original Image', image)
cv2.imshow('Shifted Image', shifted_image)

# Wait for a key press and close windows
cv2.waitKey(0)
cv2.destroyAllWindows()
```

input:

output:

6. Program to read grayscale image and zoom in or zoom out the image

```
import cv2


# Load the grayscale image

image = cv2.imread(r'C:\Users\Admin\Desktop\bunch.jpg', cv2.IMREAD_GRAYSCALE)


# Check if the image is loaded

if image is None:
```

```
    print("Error: Image not found!")
else:
    # Set the zoom scale (1.0 = original size, > 1.0 = zoom in, < 1.0 = zoom out)
    scale = 1.5  # Change this value to zoom in or out

    # Zoom the image by resizing
    zoomed_image = cv2.resize(image, (0, 0), fx=scale, fy=scale)

    # Display the original and zoomed images
    cv2.imshow('Original Image', image)
    cv2.imshow('Zoomed Image', zoomed_image)

    # Wait for a key press and close windows
    cv2.waitKey(0)
    cv2.destroyAllWindows()
```

input:

Output

7. **Program to read a grayscale image and create flipped image of original image**

```
import cv2


# Load the image in grayscale
image = cv2.imread( r'C:\Users\Admin\Desktop\bunch.jpg', cv2.IMREAD_GRAYSCALE)


# Check if the image is loaded successfully
```

```
if image is None:
    print("Error loading image.")
else:
    # Flip the image horizontally
    flipped_image = cv2.flip(image, 1)  # 1 for horizontal flip

    # Display the original and flipped images
     cv2.imshow('Original Image', image)
    cv2.imshow('Flipped Image', flipped_image)

    # Save the flipped image to a file
    cv2.imwrite('flipped_image.jpg', flipped_image)

    # Wait for a key press and close all windows
    cv2.waitKey(0)
    cv2.destroyAllWindows()
```

Input:

Output:

8. Program to read a grayscale image and blur the image

```
import cv2

# Read the image in grayscale
image = cv2.imread(r'C:\Users\Admin\Desktop\bunch.jpg', 0)

# Apply a blur effect
```

```
blurred_image = cv2.blur(image, (10, 10))

# Display the original and blurred images
cv2.imshow('Original Image', image)
cv2.imshow('Blurred Image', blurred_image)

# Wait for a key press and close the windows
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Input:

Output:

9.Program for image enhancement

```
from PIL import Image, ImageEnhance


# Load the image
image = Image.open(r"C:\Users\Admin\Desktop\bunch.jpg")

# Enhance brightness
enhancer = ImageEnhance.Brightness(image)
```

```
enhanced_image = enhancer.enhance(1.5)  # 1.0 is original, increase to make brighter


# Save the enhanced image
enhanced_image.save("enhanced_image.jpg")
enhanced_image.show()


print("Image enhancement complete!")
```

Input:

otput:

Image enhancement complete !

10. Program for image compression

```
from PIL import Image

# Load the image
image = Image.open(r"C:\Users\Admin\Desktop\bunch.jpg")

# Compress the image and save it with reduced quality
compressed_image_path = "compressed_image.jpg"
image.save(compressed_image_path, "JPEG", quality=10)  # Change quality as needed
```

```
# Show the compressed image
compressed_image = Image.open(compressed_image_path)
compressed_image.show()

print("Image compressed")
```

Input image:

Output :

```
Image compressed
```

11. Program for image segmentation

```
import cv2

import matplotlib.pyplot as plt
```

```
def segment_image(image_path, threshold_value=127):
    # Read the image in grayscale
    image = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)

    # Apply binary thresholding
    _, segmented_image = cv2.threshold(image, threshold_value, 255, cv2.THRESH_BINARY)

    # Display original and segmented images
    plt.figure(figsize=(10, 5))
    plt.subplot(1, 2, 1)
    plt.title('Original Image')
    plt.imshow(image, cmap='gray')
    plt.axis('off')

    plt.subplot(1, 2, 2)
    plt.title('Segmented Image')
    plt.imshow(segmented_image, cmap='gray')
    plt.axis('off')

    plt.show()

# Example usage
```

segment_image(r'C:\Users\Admin\Desktop\bunch.jpg', threshold_value=127)

Input image:

Output: