```
!pip install imageio
```

```
Requirement already satisfied: imageio in /usr/local/lib/python3.11/dist-packages (2.37.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (from imageio) (1.26.4)
Requirement already satisfied: pillow>=8.3.2 in /usr/local/lib/python3.11/dist-packages (from imageio) (11.1.0)
```

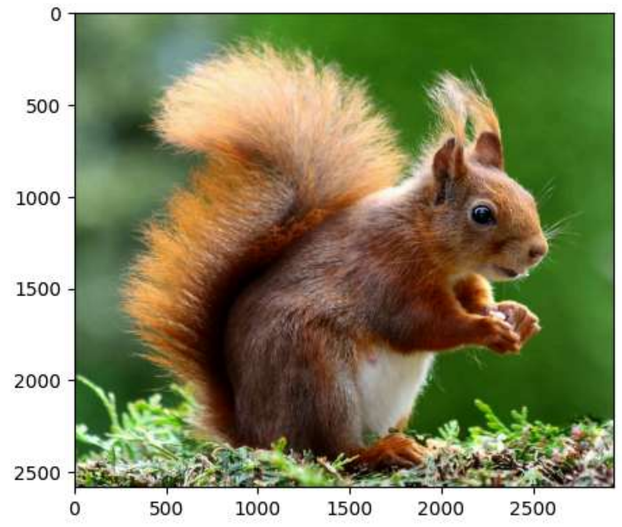
```
!pip install numpy
```

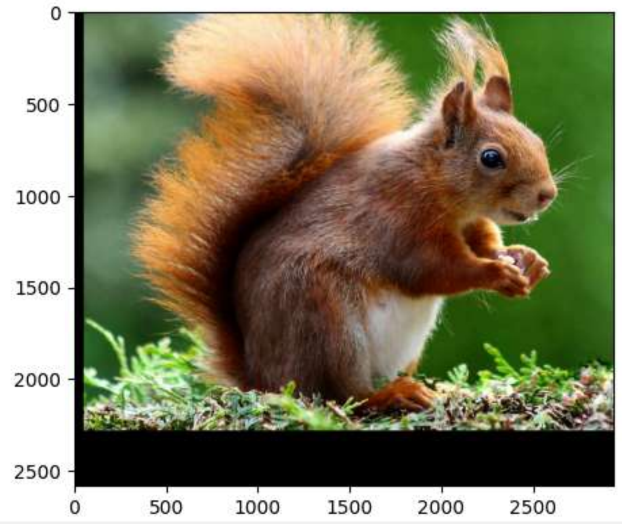
```
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (1.26.4)
```

```
!pip install matplotlib
```

```
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages (3.10.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.3.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (4.56.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.4.8)
Requirement already satisfied: numpy>=1.23 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.26.4)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (24.2)
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (3.2.1)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.7->matplotlib) (1.17.0)
```

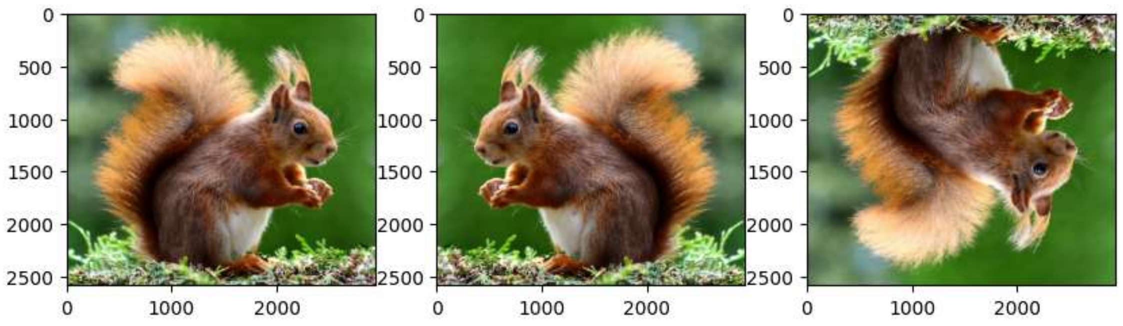
```
import imageio.v3 as img
import numpy as np
import matplotlib.pyplot as plt

def Translasi(image, shiftX, shiftY):
    imgTranslasi = np.roll(image, shift=shiftY, axis=0) # Geser vertikal
    imgTranslasi = np.roll(imgTranslasi, shift=shiftX, axis=1) # Geser horizontal
    # Mengisi bagian yang kosong dengan warna hitam (0)
    if shiftY > 0:
        imgTranslasi[:shiftY, :] = 0 # Bagian atas jika geser ke bawah
    elif shiftY < 0:
        imgTranslasi[shiftY:, :] = 0 # Bagian bawah jika geser ke atas
    if shiftX > 0:
        imgTranslasi[:, :shiftX] = 0 # Bagian kiri jika geser ke kanan
    elif shiftX < 0:
        imgTranslasi[:, shiftX:] = 0 # Bagian kanan jika geser ke kiri
    return imgTranslasi

image = img.imread("sampel1.jpg")
imgResult = Translasi(image, shiftX=50, shiftY=-300)

plt.figure(figsize=(10,10))
plt.subplot(2,1,1)
plt.imshow(image)
plt.subplot(2,1,2)
plt.imshow(imgResult)
plt.show()
```





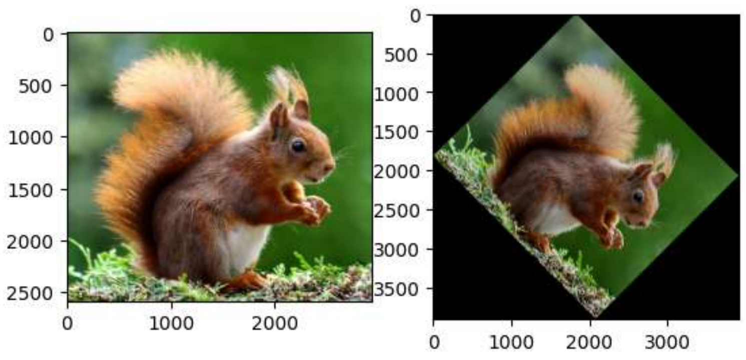
```
import numpy as np
import imageio as img
import matplotlib.pyplot as plt

path = 'sampel1.jpg'
image = img.imread(path)
height, width = image.shape[:2]
horizontal = np.zeros_like(image)
vertical = np.zeros_like(image)
for y in range(height):
  for x in range(width):
    horizontal[y, x] = image[y, width - 1 - x]
for y in range(height):
  for x in range(width):
    vertical[y, x] = image[height - 1 - y, x]
plt.figure(figsize=(10, 5))
plt.subplot(1, 3, 1)
plt.imshow(image)
plt.subplot(1, 3, 2)
plt.imshow(horizontal)
plt.subplot(1, 3, 3)
plt.imshow(vertical)
plt.show()
```

```
<ipython-input-11-f9d51152fed6>:6: DeprecationWarning: Starting with ImageIO v3 the behavior of this function will switch to that of i
  image = img.imread(path)
```



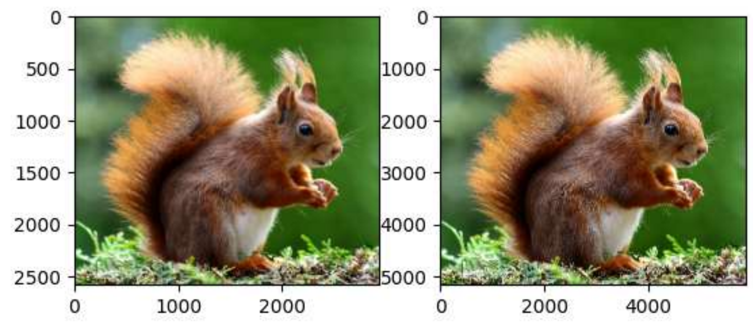
```
import imageio as img
import numpy as np
import matplotlib.pyplot as plt

def rotateImage(image, degree):
  radian_deg = np.radians(degree)
  cos_deg, sin_deg = np.cos(radian_deg), np.sin(radian_deg)
  height, width = image.shape[:2]
  max_dim = int(np.sqrt(height**2 + width**2))
  outputImage = np.zeros((max_dim, max_dim, 3), dtype=image.dtype)
  centerY, centerX = max_dim//2, max_dim//2
  for y in range(-height//2, height//2):
    for x in range(-width//2, width//2):
      newX = int(cos_deg * x - sin_deg * y) + centerX
      newY = int(sin_deg * x + cos_deg * y) + centerY
      if 0 <= newX < max_dim and 0 <= newY < max_dim:
        outputImage[newY, newX] = image[y + height//2, x + width//2]
  return outputImage

image = img.imread('sampel1.jpg')
rotated_image = rotateImage(image, 45)

plt.subplot(1, 2, 1)
plt.imshow(image)

plt.subplot(1, 2, 2)
plt.imshow(rotated_image)

plt.show()
```

```
<ipython-input-13-a0349a8360d4>:20: DeprecationWarning: Starting with ImageIO v3 the behavior of this function will switch to that of
  image = img.imread('sampel1.jpg')
```

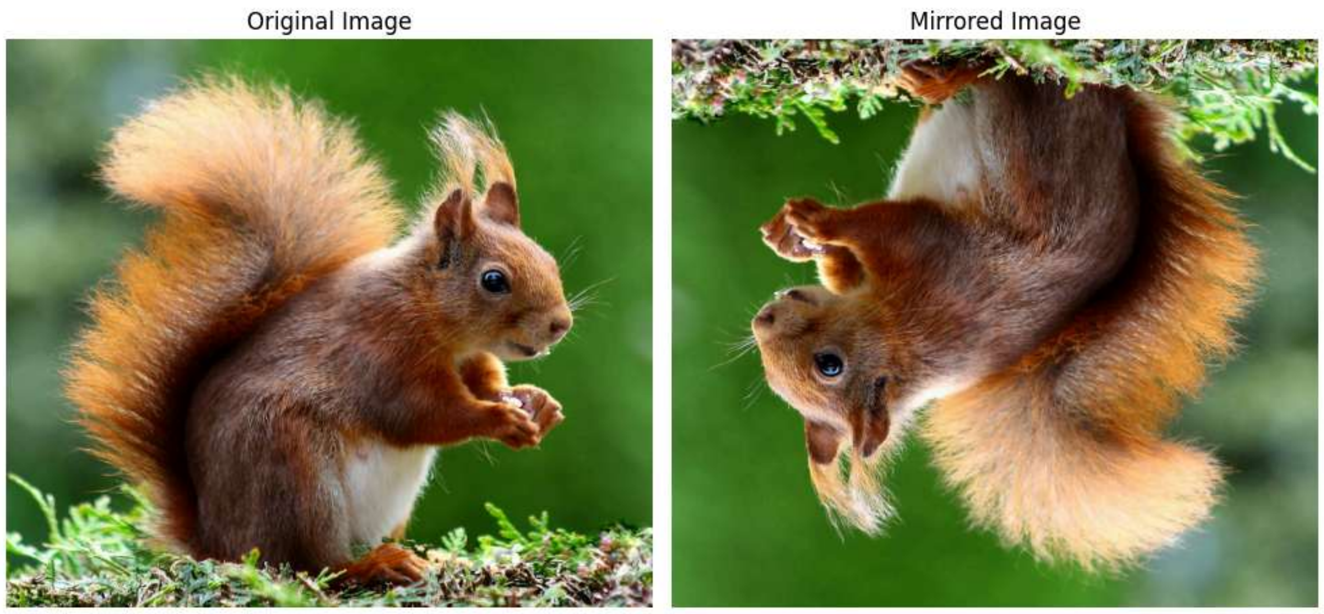


```
import numpy as np
import imageio as img
import matplotlib.pyplot as plt

def zoomPlus(image, factor):
  height, width = image.shape[:2]
  new_height = int(height * factor)
  new_width = int(width * factor)
  imgZoom = np.zeros((new_height, new_width, 3), dtype=image.dtype)
  for y in range(new_height):
    for x in range(new_width):
      ori_y = int(y / factor)
      ori_x = int(x / factor)
      ori_y = min(ori_y, height - 1)
      ori_x = min(ori_x, width - 1)
      imgZoom[y, x] = image[ori_y, ori_x]
  return imgZoom

image = img.imread('sampel1.jpg')
skala = 2.0

imgZoom = zoomPlus(image, skala)
img.imwrite("hasil.jpg", imgZoom)
plt.subplot(1, 2, 1)
plt.imshow(image)

plt.subplot(1, 2, 2)
plt.imshow(imgZoom)
plt.show()
```

```
<ipython-input-25-ea7361bb6b54>:19: DeprecationWarning: Starting with ImageIO v3 the behavior of this function will switch to that of
  image = img.imread('sampel1.jpg')
```

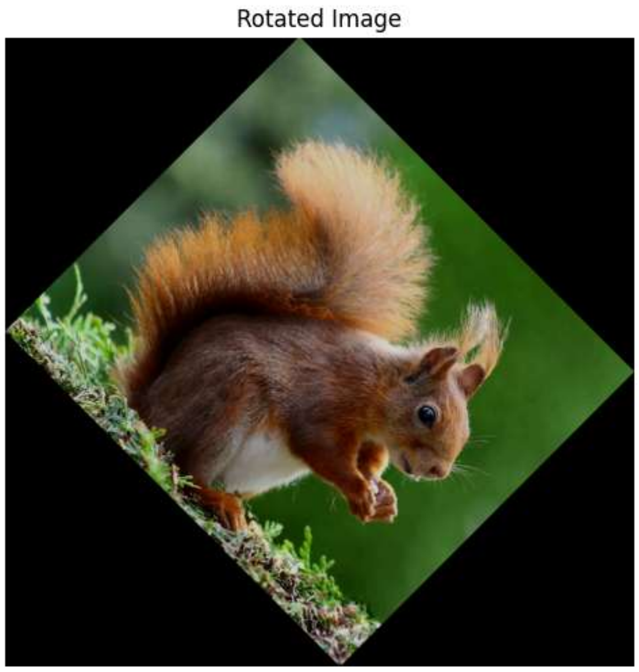


```
import numpy as np
import imageio.v3 as img
import matplotlib.pyplot as plt

def mirror_vertical_horizontal(image):
    # Mirroring vertikal
    mirrored_vertical = np.flipud(image)
    # Mirroring horizontal
    mirrored_both = np.fliplr(mirrored_vertical)
    return mirrored_both

# Baca citra
image_path = 'sampel1.jpg'  # Ganti dengan path citra Anda
original_image = img.imread(image_path)

# Lakukan mirroring vertikal dan horizontal
mirrored_image = mirror_vertical_horizontal(original_image)

# Tampilkan citra asli dan citra yang dimirroring
plt.figure(figsize=(10, 5))

plt.subplot(1, 2, 1)
plt.title('Original Image')
plt.imshow(original_image)
plt.axis('off')

plt.subplot(1, 2, 2)
plt.title('Mirrored Image')
plt.imshow(mirrored_image)
plt.axis('off')

plt.tight_layout()
plt.show()
```



```
import numpy as np
import imageio as img
import matplotlib.pyplot as plt

def rotate_image(image, angle):
    # Menghitung sudut dalam radian
    angle_rad = np.deg2rad(angle)

    # Mendapatkan dimensi citra
    height, width = image.shape[:2]

    # Menghitung dimensi citra baru setelah rotasi
    new_height = int(abs(height * np.cos(angle_rad)) + abs(width * np.sin(angle_rad)))
    new_width = int(abs(width * np.cos(angle_rad)) + abs(height * np.sin(angle_rad)))

    # Membuat citra baru dengan latar belakang hitam
    rotated_image = np.zeros((new_height, new_width, 3), dtype=image.dtype)

    # Menghitung titik tengah dari citra baru
    center_x, center_y = new_width // 2, new_height // 2

    # Melakukan rotasi
    for y in range(height):
        for x in range(width):
            # Menghitung koordinat baru setelah rotasi
            new_x = int(center_x + (x - width / 2) * np.cos(angle_rad) - (y - height / 2) * np.sin(angle_rad))
            new_y = int(center_y + (x - width / 2) * np.sin(angle_rad) + (y - height / 2) * np.cos(angle_rad))

            # Memastikan koordinat baru berada dalam batas citra
            if 0 <= new_x < new_width and 0 <= new_y < new_height:
                rotated_image[new_y, new_x] = image[y, x]

    return rotated_image

# Membaca citra
image_path = 'sampel1.jpg'  # Ganti dengan path citra Anda
original_image = img.imread(image_path)

# Melakukan rotasi
angle = 45  # Ganti dengan sudut rotasi yang diinginkan
rotated_image = rotate_image(original_image, angle)

# Menampilkan citra asli dan citra yang sudah dirotasi
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.title('Original Image')
plt.imshow(original_image)
plt.axis('off')

plt.subplot(1, 2, 2)
plt.title('Rotated Image')
plt.imshow(rotated_image)
plt.axis('off')

plt.tight_layout()
plt.show()
```

```
<ipython-input-27-e8b7baea72a8>:37: DeprecationWarning: Starting with ImageIO v3 the behavior of this function will switch to that of
  original_image = img.imread(image_path)
```



```
import numpy as np
import imageio as img
import matplotlib.pyplot as plt

def zoomMinus(image, factor):
    """
    Memperkecil gambar menggunakan interpolasi tetangga terdekat.

    Parameters:
    image (numpy.ndarray): Gambar input dalam format array NumPy.
    factor (float): Faktor pengurangan ukuran gambar (harus lebih dari 1).

    Returns:
    numpy.ndarray: Gambar yang telah diperkecil.
    """
    if factor <= 1:
        raise ValueError("Faktor pengurangan harus lebih dari 1.")

    height, width = image.shape[:2]
    new_height = int(height / factor)
    new_width = int(width / factor)
    imgZoom = np.zeros((new_height, new_width, 3), dtype=image.dtype)

    for y in range(new_height):
        for x in range(new_width):
            ori_y = int(y * factor)
            ori_x = int(x * factor)
            ori_y = min(ori_y, height - 1)
            ori_x = min(ori_x, width - 1)
            imgZoom[y, x] = image[ori_y, ori_x]

    return imgZoom

# Contoh penggunaan
image = img.imread('sampel1.jpg')
skala = 2.0  # Faktor pengurangan
```

```
    imgZoom = zoomMinus(image, skala)
    img.imwrite("hasil_zoom_minus.jpg", imgZoom)

    plt.figure(figsize=(10, 5))
    plt.subplot(1, 2, 1)
    plt.title('Gambar Asli')
    plt.imshow(image)

    plt.subplot(1, 2, 2)
    plt.title('Gambar yang Diperkecil')
    plt.imshow(imgZoom)

    plt.tight_layout()
    plt.show()
```

```
<ipython-input-28-4f5578a626c4>:35: DeprecationWarning: Starting with ImageIO v3 the behavior of this function will switch to that of
  image = img.imread('sampel1.jpg')
```