João Eduardo **BI3007588**

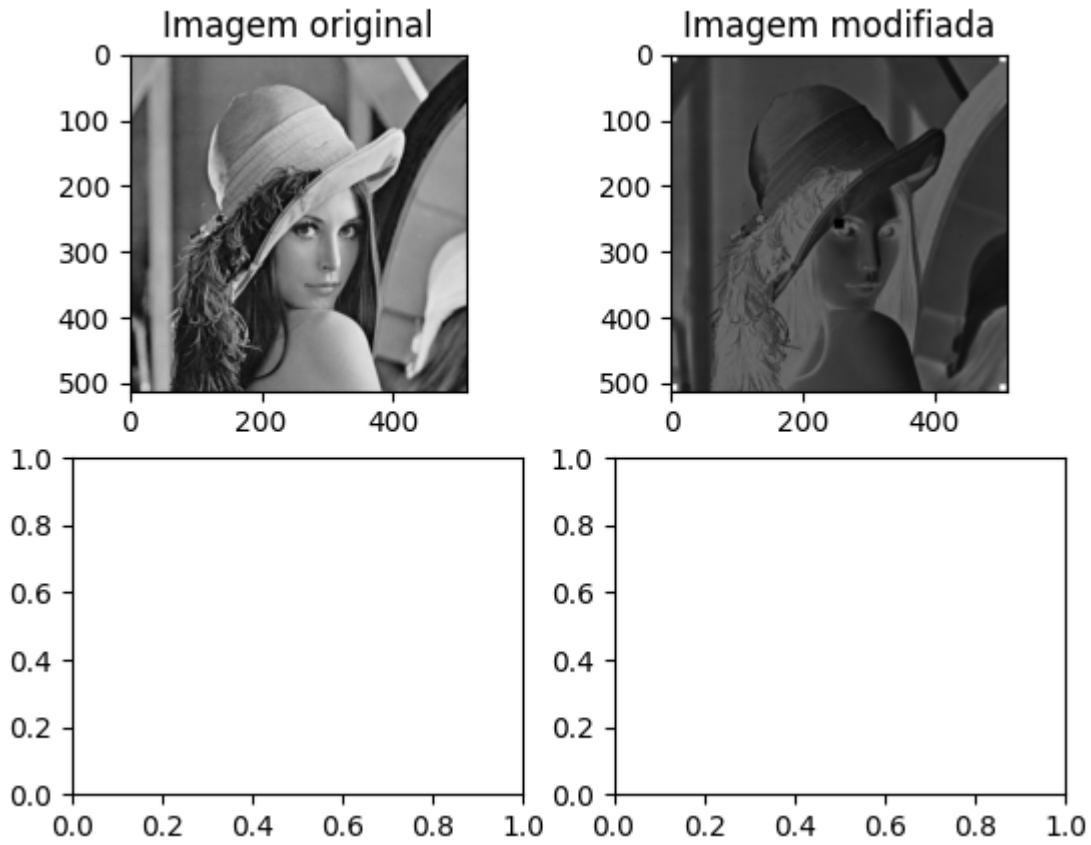
Lista PDI

[OPERAÇÃO PONTO A PONTO]

Lena:

```
#Exemplo de leitura e plot de imagens
#Lembrar de adicionar as bibiotecas: Numpy, Pillow, Matplotlib

import numpy as np
from PIL import Image # pillow
import matplotlib.pyplot as plt

def main():
    img = Image.open('lena_gray_512.tif')

    #img.show() # Plot image using Pillow
    # converte image to numpy array
    npImg = np.array(img)

    #Calcular o negativo das imagens
    npImg = 255-npImg

    #Diminuir pela metade a intensidade dos pixels
    npImg = npImg/2

    #Incluir 4 quadrados brancos 10 x 10 pixels em cada canto das
imagens
    npImg[0:10,0:10] = 255
    npImg[501:511,501:511] = 255
    npImg[501:511,0:10] = 255
    npImg[0:10,501:511] = 255

    #Incluir 1 quadrado preto 15X15 no centro das imagens
    npImg[249:264,249:264] = 0

    #Converte numpy array to Image
    imgNew = Image.fromarray(npImg)
    #imgNew.show()
    # Plot using matplotlib
    fig, ax = plt.subplots(nrows=2, ncols=2)
```

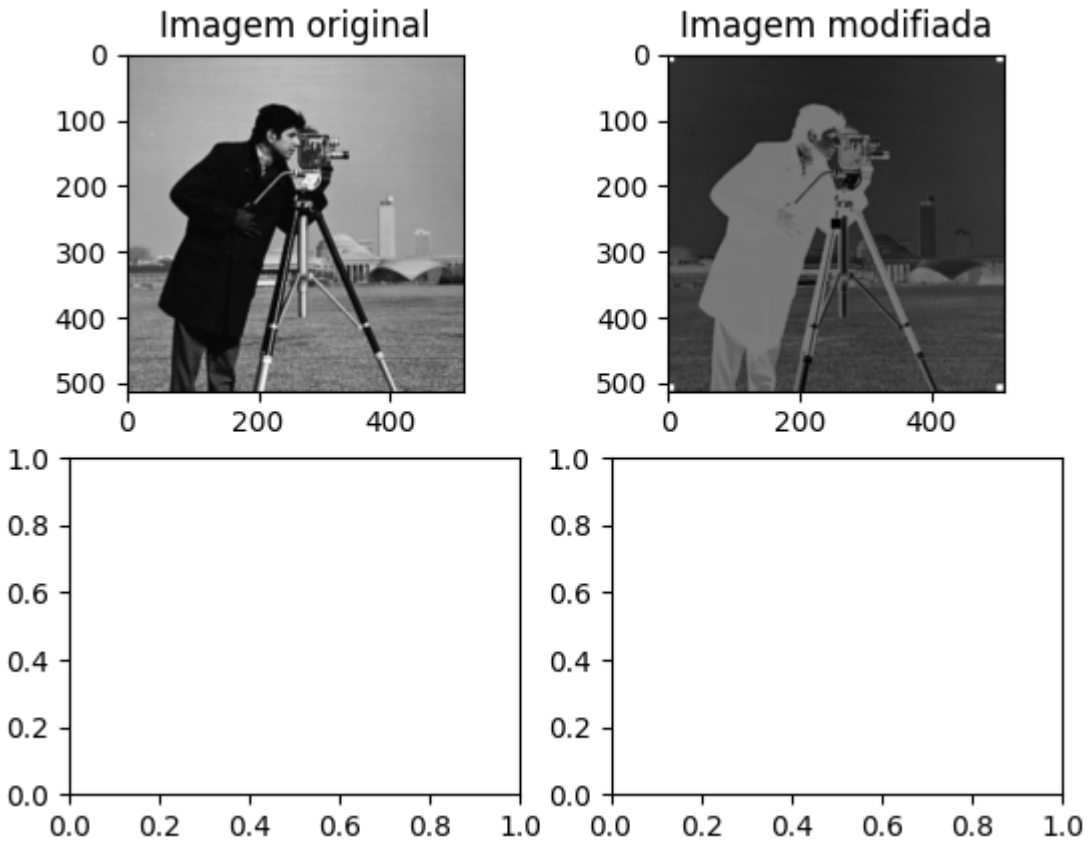
```
    ax[0,0].imshow(img, cmap='gray')
    ax[0,0].set_title("Imagem original")
    ax[0,1].imshow(imgNew, cmap='gray')
    ax[0,1].set_title("Imagem modificada")
    plt.show()

if __name__ == "__main__":
    main()
```



Cameraman:

```
#Exemplo de leitura e plot de imagens
#Lembrar de adicionar as bibiotecas: Numpy, Pillow, Matplotlib

import numpy as np
from PIL import Image # pillow
import matplotlib.pyplot as plt

def main():
    img = Image.open('cameraman.tif')
```

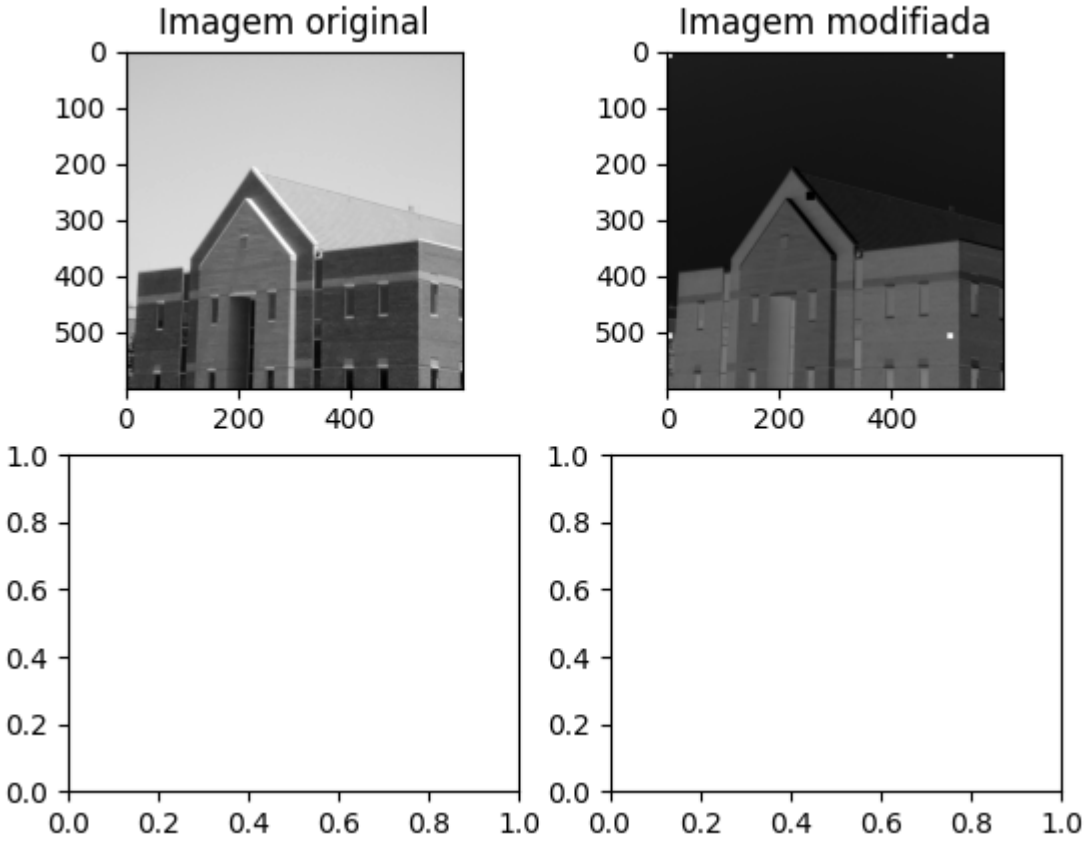
```
    #img.show() # Plot image using Pillow
    # converte image to numpy array
    npImg = np.array(img)

    #Calcular o negativo das imagens
    npImg = 255-npImg

    #Diminuir pela metade a intensidade dos pixels
    npImg = npImg/2

    #Incluir 4 quadrados brancos 10 x 10 pixels em cada canto das
imagens
    npImg[0:10,0:10] = 255
    npImg[501:511,501:511] = 255
    npImg[501:511,0:10] = 255
    npImg[0:10,501:511] = 255

    #Incluir 1 quadrado preto 15X15 no centro das imagens
    npImg[249:264,249:264] = 0

    #Converte numpy array to Image
    imgNew = Image.fromarray(npImg)
    #imgNew.show()
    # Plot using matplotlib
    fig, ax = plt.subplots(nrows=2, ncols=2)
    ax[0,0].imshow(img, cmap='gray')
    ax[0,0].set_title("Imagem original")
    ax[0,1].imshow(imgNew, cmap='gray')
    ax[0,1].set_title("Imagem modificada")
    plt.show()

if __name__ == "__main__":
    main()
```



House:

```
#Exemplo de leitura e plot de imagens
#Lembrar de adicionar as bibiotecas: Numpy, Pillow, Matplotlib

import numpy as np
from PIL import Image # pillow
import matplotlib.pyplot as plt

def main():
    img = Image.open('house.tif')

    #img.show() # Plot image using Pillow
    # converte image to numpy array
    npImg = np.array(img)

    #Calcular o negativo das imagens
    npImg = 255-npImg

    #Diminuir pela metade a intensidade dos pixels
```

```
    npImg = npImg/2

    #Incluir 4 quadrados brancos 10 x 10 pixels em cada canto das
imagens
    npImg[0:10,0:10] = 255
    npImg[501:511,501:511] = 255
    npImg[501:511,0:10] = 255
    npImg[0:10,501:511] = 255

    #Incluir 1 quadrado preto 15X15 no centro das imagens
    npImg[249:264,249:264] = 0

    #Converte numpy array to Image
    imgNew = Image.fromarray(npImg)
    #imgNew.show()
    # Plot using matplotlib
    fig, ax = plt.subplots(nrows=2, ncols=2)
    ax[0,0].imshow(img, cmap='gray')
    ax[0,0].set_title("Imagem original")
    ax[0,1].imshow(imgNew, cmap='gray')
    ax[0,1].set_title("Imagem modificada")
    plt.show()

if __name__ == "__main__":
    main()
```

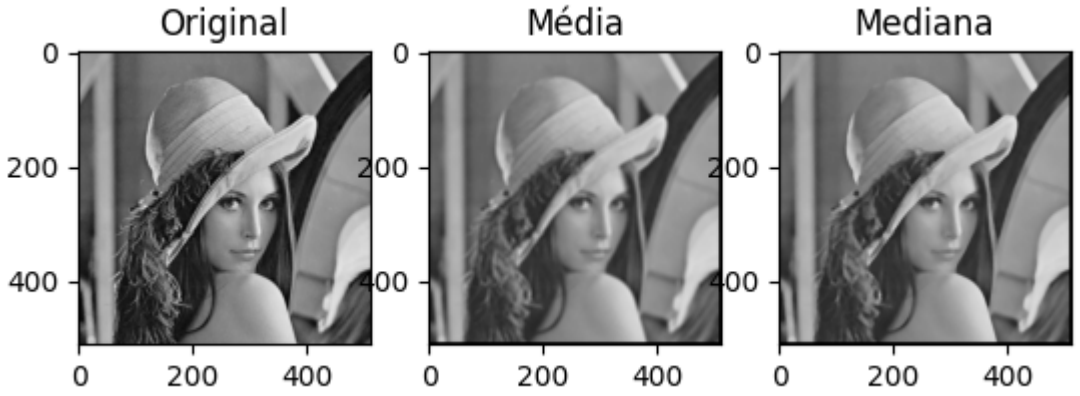


[OPERAÇÃO POR VIZINHANÇA]

Lena:

```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt

def main():
    image_pillow = Image.open('lena_gray_512.tif')
    f_image_nd = np.array(image_pillow)
    g_image_nd = np.zeros(f_image_nd.shape)
    p_image_nd = np.zeros(f_image_nd.shape)

    # Operação por vizinhaça
    l = f_image_nd.shape[0]
    c = f_image_nd.shape[1]
    k = 3 #kernel size

    for x in range(k, l-k):
        for y in range(k, c-k):
            s_xy = f_image_nd[x-k:x+k+1, y-k:y+k+1]
```

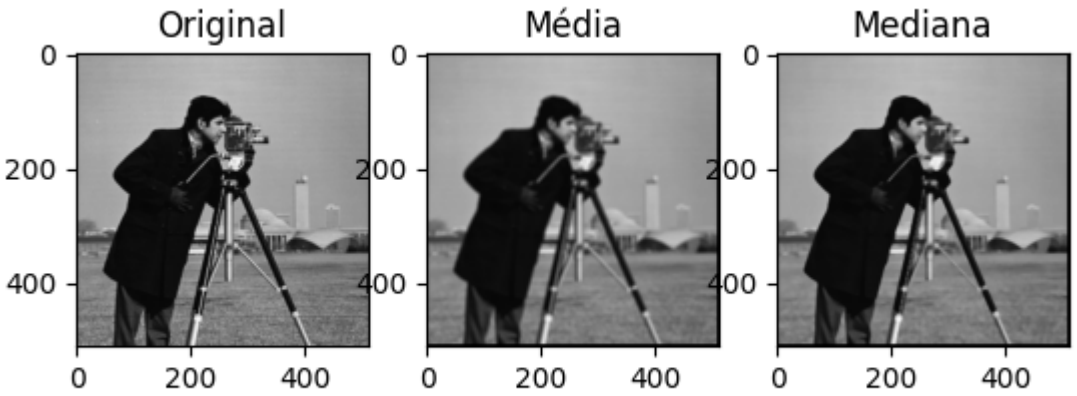
```
            g_image_nd[x,y] = np.mean(s_xy).astype(int)
            p_image_nd[x,y] = np.median(s_xy).astype(int)

    #Criando colunas para plotagem
    fig = plt.figure()
    plt1 = plt.subplot(1,3,1)
    plt2 = plt.subplot(1,3,2)
    plt3 = plt.subplot(1,3,3)

    plt1.title.set_text('Original')
    plt1.imshow(f_image_nd, cmap='gray')

    plt2.title.set_text('Média')
    plt2.imshow(g_image_nd, cmap='gray', vmin=0, vmax=255)

    plt3.title.set_text('Mediana')
    plt3.imshow(p_image_nd, cmap='gray', vmin=0, vmax=255)

    plt.show()

if __name__ == "__main__":
    main()
```



Cameraman:

```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt

def main():
    image_pillow = Image.open('cameraman.tif')
    f_image_nd = np.array(image_pillow)
    g_image_nd = np.zeros(f_image_nd.shape)
```

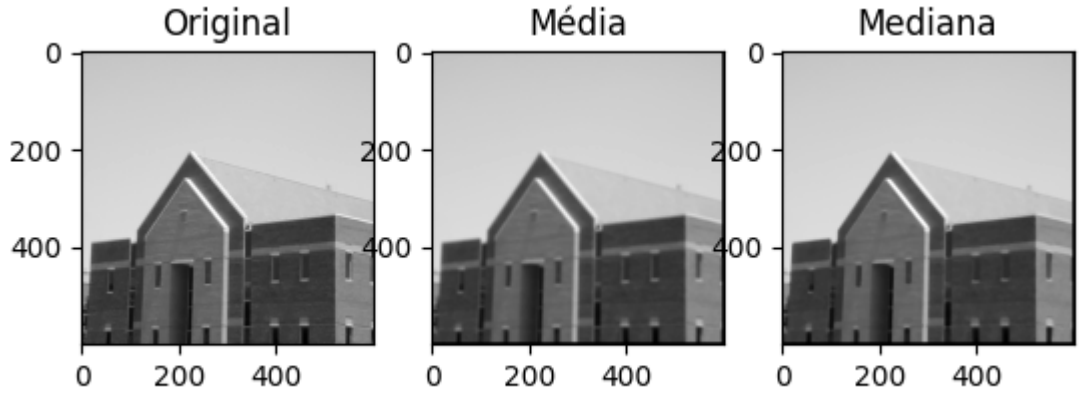
```
    p_image_nd = np.zeros(f_image_nd.shape)

    # Operação por vizinhaça
    l = f_image_nd.shape[0]
    c = f_image_nd.shape[1]
    k = 3 #kernel size

    for x in range(k, l-k):
        for y in range(k, c-k):
            s_xy = f_image_nd[x-k:x+k+1, y-k:y+k+1]
            g_image_nd[x,y] = np.mean(s_xy).astype(int)
            p_image_nd[x,y] = np.median(s_xy).astype(int)

    #Criando colunas para plotagem
    fig = plt.figure()
    plt1 = plt.subplot(1,3,1)
    plt2 = plt.subplot(1,3,2)
    plt3 = plt.subplot(1,3,3)

    plt1.title.set_text('Original')
    plt1.imshow(f_image_nd, cmap='gray')

    plt2.title.set_text('Média')
    plt2.imshow(g_image_nd, cmap='gray', vmin=0, vmax=255)

    plt3.title.set_text('Mediana')
    plt3.imshow(p_image_nd, cmap='gray', vmin=0, vmax=255)

    plt.show()

if __name__ == "__main__":
    main()
```



House

```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt

def main():
    image_pillow = Image.open('house.tif')
    f_image_nd = np.array(image_pillow)
    g_image_nd = np.zeros(f_image_nd.shape)
    p_image_nd = np.zeros(f_image_nd.shape)

    # Operação por vizinhaça
    l = f_image_nd.shape[0]
    c = f_image_nd.shape[1]
    k = 3 #kernel size

    for x in range(k, l-k):
        for y in range(k, c-k):
            s_xy = f_image_nd[x-k:x+k+1, y-k:y+k+1]
            g_image_nd[x,y] = np.mean(s_xy).astype(int)
            p_image_nd[x,y] = np.median(s_xy).astype(int)

    #Criando colunas para plotagem
    fig = plt.figure()
    plt1 = plt.subplot(1,3,1)
    plt2 = plt.subplot(1,3,2)
    plt3 = plt.subplot(1,3,3)

    plt1.title.set_text('Original')
    plt1.imshow(f_image_nd, cmap='gray')
```

```
    plt2.title.set_text('Média')
    plt2.imshow(g_image_nd, cmap='gray', vmin=0, vmax=255)

    plt3.title.set_text('Mediana')
    plt3.imshow(p_image_nd, cmap='gray', vmin=0, vmax=255)

    plt.show()

if __name__ == "__main__":
    main()
```



[TRANSFORMAÇÕES GEOMÉTRICAS] - ZOOM

Lena:

```
#TRANSFORMAÇÕES GEOMÉTRICAS
import numpy as np
from numpy import asarray
from PIL import Image
from scipy import ndimage

def main():
    #Selecionar imagem
    image_in = Image.open('lena_gray_512.tif')
    image_in.show()

    #Conversão imagem array
    image_np = np.array(image_in)
    print(image_np.shape)

    #Zoom
    image_np_zoom = ndimage.zoom(image_np, (2.5, 2.5))
    print(image_np_zoom.shape)
```

```
    image_out = Image.fromarray(image_np_zoom)
    image_out.show()

if __name__ == "__main__":
    main()
```

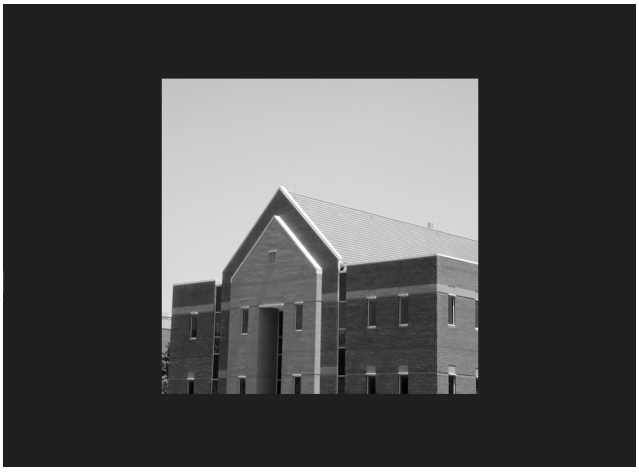
Cameraman:

```
#TRANSFORMAÇÕES GEOMÉTRICAS
import numpy as np
from numpy import asarray
from PIL import Image
from scipy import ndimage

def main():
    #Selecionar imagem
    image_in = Image.open('cameraman.tif')
    image_in.show()

    #Conversão imagem array
    image_np = np.array(image_in)
    print(image_np.shape)

    #Zoom
    image_np_zoom = ndimage.zoom(image_np, (2.5, 2.5))
    print(image_np_zoom.shape)

    image_out = Image.fromarray(image_np_zoom)
    image_out.show()
```

```
if __name__ == "__main__":
    main()
```

House:

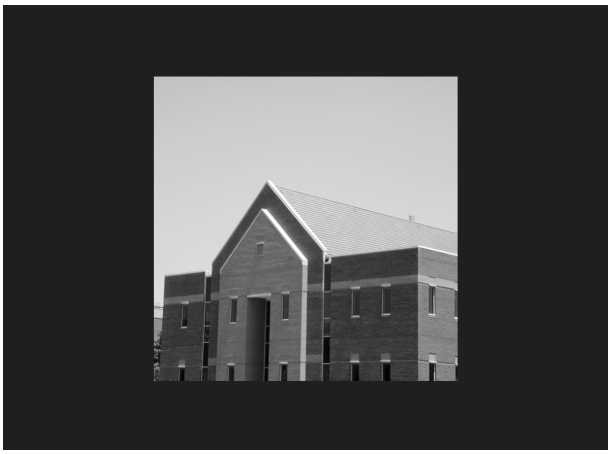
```
#TRANSFORMAÇÕES GEOMÉTRICAS
import numpy as np
from numpy import asarray
from PIL import Image
from scipy import ndimage

def main():
    #Selecionar imagem
    image_in = Image.open('house.tif')
    image_in.show()

    #Conversão imagem array
    image_np = np.array(image_in)
    print(image_np.shape)

    #Zoom
    image_np_zoom = ndimage.zoom(image_np, (2.5, 2.5))
    print(image_np_zoom.shape)

    image_out = Image.fromarray(image_np_zoom)
    image_out.show()

if __name__ == "__main__":
    main()
```

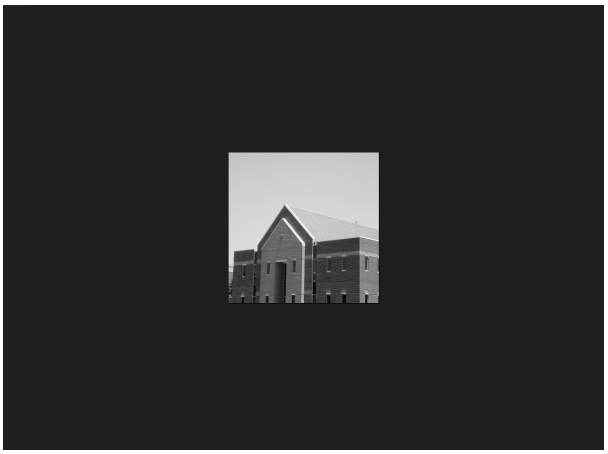


[TRANSFORMAÇÕES GEOMÉTRICAS] - ZOOM OUT

House:

```
# Image Geometric transforms
import numpy as np
from numpy import asarray
from PIL import Image
from scipy import ndimage

def main():
    # Open image
    image_in = Image.open('house.tif')
    image_in.show()
    # convert image to numpy array
    image_np = np.array(image_in)
    print(image_np.shape)

    # Zoom or Shrink image
    image_np_zoom = ndimage.zoom(image_np, (0.5, 0.5))
    print(image_np_zoom.shape)
    image_out = Image.fromarray(image_np_zoom)
    image_out.show()

if __name__ == "__main__":
    main()
```





Lena:

```
# Image Geometric transforms
import numpy as np
from numpy import asarray
from PIL import Image
from scipy import ndimage

def main():
    # Open image
    image_in = Image.open('lena_gray_512.tif')
    image_in.show()
    # convert image to numpy array
    image_np = np.array(image_in)
    print(image_np.shape)

    # Zoom or Shrink image
    image_np_zoom = ndimage.zoom(image_np, (0.5, 0.5))
    print(image_np_zoom.shape)
    image_out = Image.fromarray(image_np_zoom)
    image_out.show()

if __name__ == "__main__":
    main()
```

Cameraman:

```
# Image Geometric transforms
import numpy as np
from numpy import asarray
from PIL import Image
from scipy import ndimage

def main():
    # Open image
    image_in = Image.open('cameraman.tif')
    image_in.show()
    # convert image to numpy array
    image_np = np.array(image_in)
    print(image_np.shape)

    # Zoom or Shrink image
    image_np_zoom = ndimage.zoom(image_np, (0.5, 0.5))
    print(image_np_zoom.shape)
    image_out = Image.fromarray(image_np_zoom)
    image_out.show()

if __name__ == "__main__":
    main()
```

[TRANSFORMAÇÕES GEOMÉTRICAS]: Rotação em 45º, 90º e 100º

```
from PIL import Image

#read the image
im = Image.open('cameraman.tif')

#rotate image
angulo = 45
out = im.rotate(angulo)

out.save('imagemalterada.png')
```

-O código para todas imagens é esse, alterando somente o caminho da imagem em questão e a angulação desejada em “angulo”. Representarei os resultados usando o cameraman.

[TRANSFORMAÇÕES GEOMÉTRICAS]: Translação para as coordenadas desejadas

```
from PIL import Image
import PIL.ImageOps
import glob

files = glob.glob('cameraman.tif')

for f in files:
    image = Image.open(f)
    inverted_image = PIL.ImageOps.invert(image)

    width, height = image.size
    shifted_image = Image.new("RGB", (width+35, height))
    shifted_image.paste(image, (35, 45))
    shifted_image.save('Imagem alterada.jpg' )
```

-O código para todas imagens é esse, alterando somente o caminho da imagem em questão e os pixels em que o usuário deseja(penúltima e antepenúltima linha). Representarei o resultado usando o cameraman.