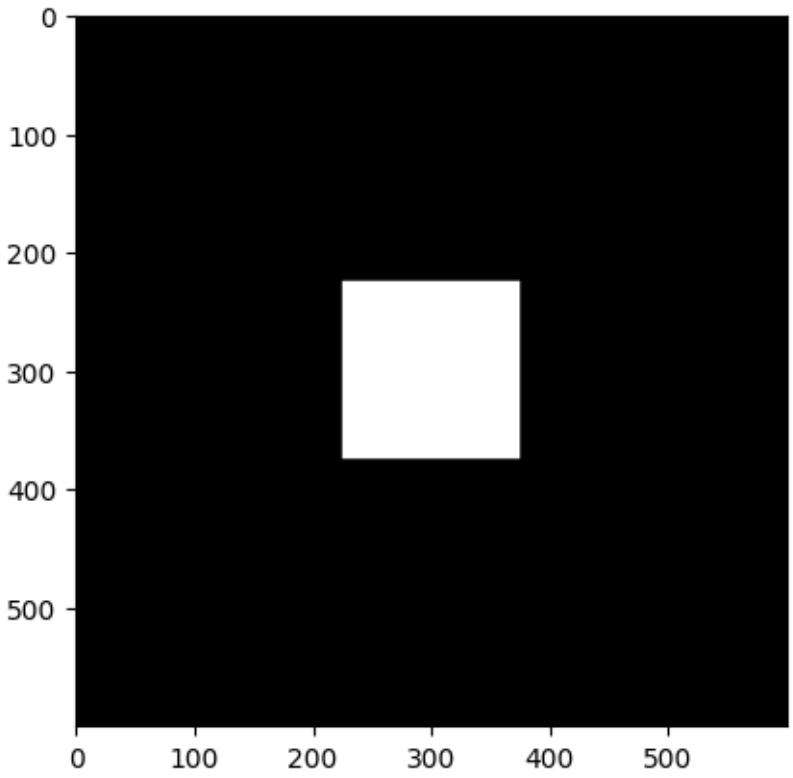
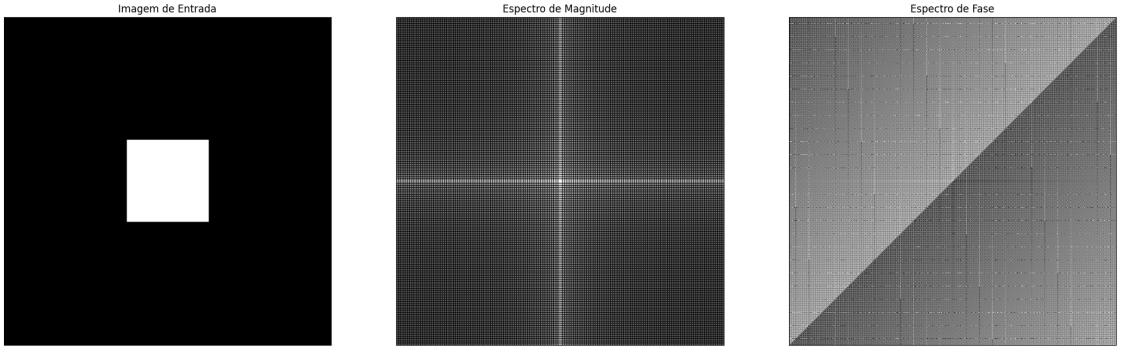
```
        plt1.set_title('Imagem Original')
        plt1.set_xticks([]), plt1.set_yticks([])
        plt1.imshow(original_img, cmap='gray')

        # Subplot 2: Imagem reconstruída a partir do espectro de magnitude
        plt2 = plt.subplot(1, 2, 2)
        plt2.set_title('Imagem Reconstruída')
        plt2.set_xticks([]), plt2.set_yticks([])
        plt2.imshow(inverse_transformed_image, cmap='gray')

        # Exibe a figura com as duas imagens
        plt.show()


    return inverse_transformed_image

i = 0
for img in [img_car, img_lena_periodic_noise, img_newspaper_shot_woman,␣
 ↪img_periodic_noise, img_sinc]:
    inverse_fourier_img = apply_inverse_fourier_transform(img,fourier_img[i][2])
    i+=1
```

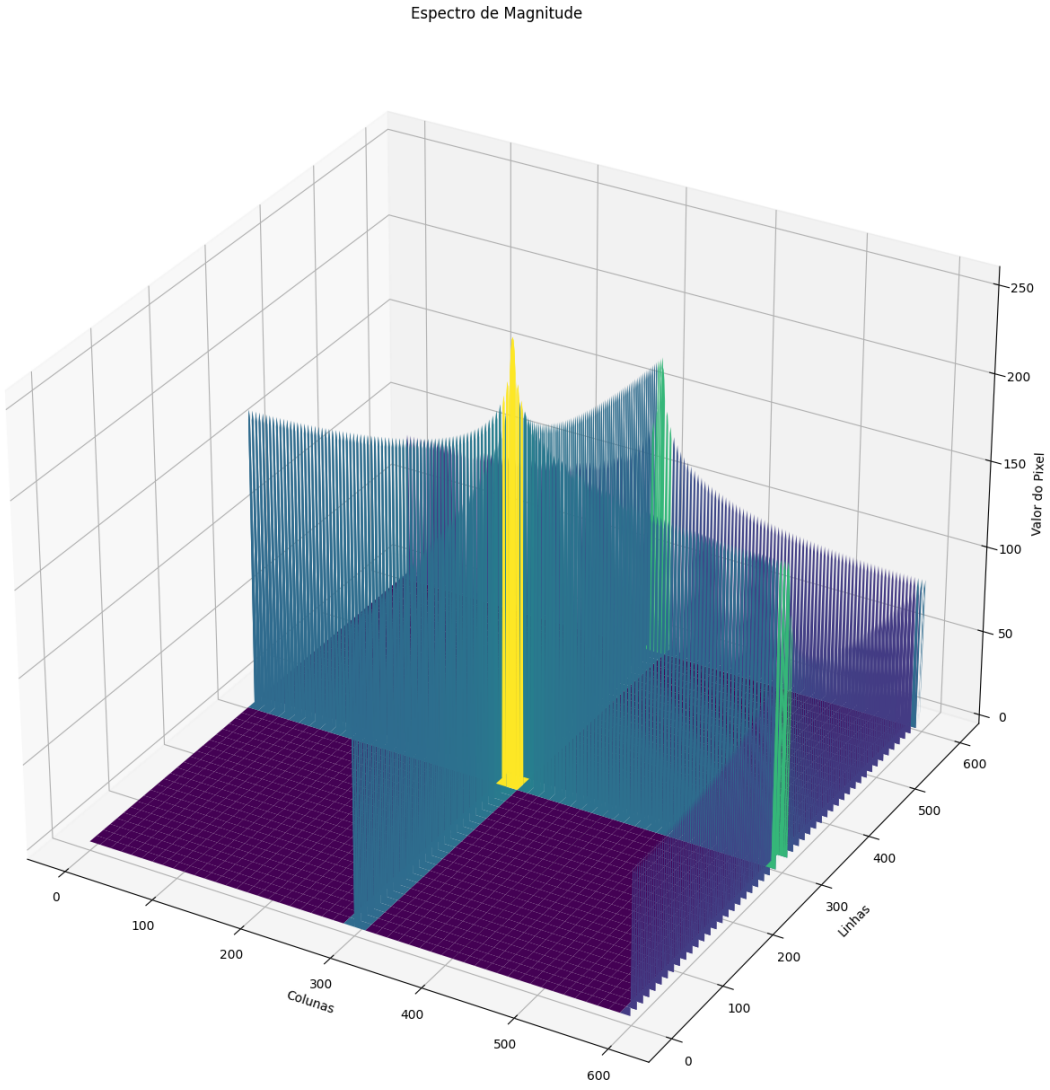
O código a seguir aplica a Transformada de Fourier Discreta (DFT) e a Transformada Inversa de Fourier em uma imagem com fundo preto e quadrado branco no centro representando a função sinc(x,y) e exibe os resultados plotando as imagens.

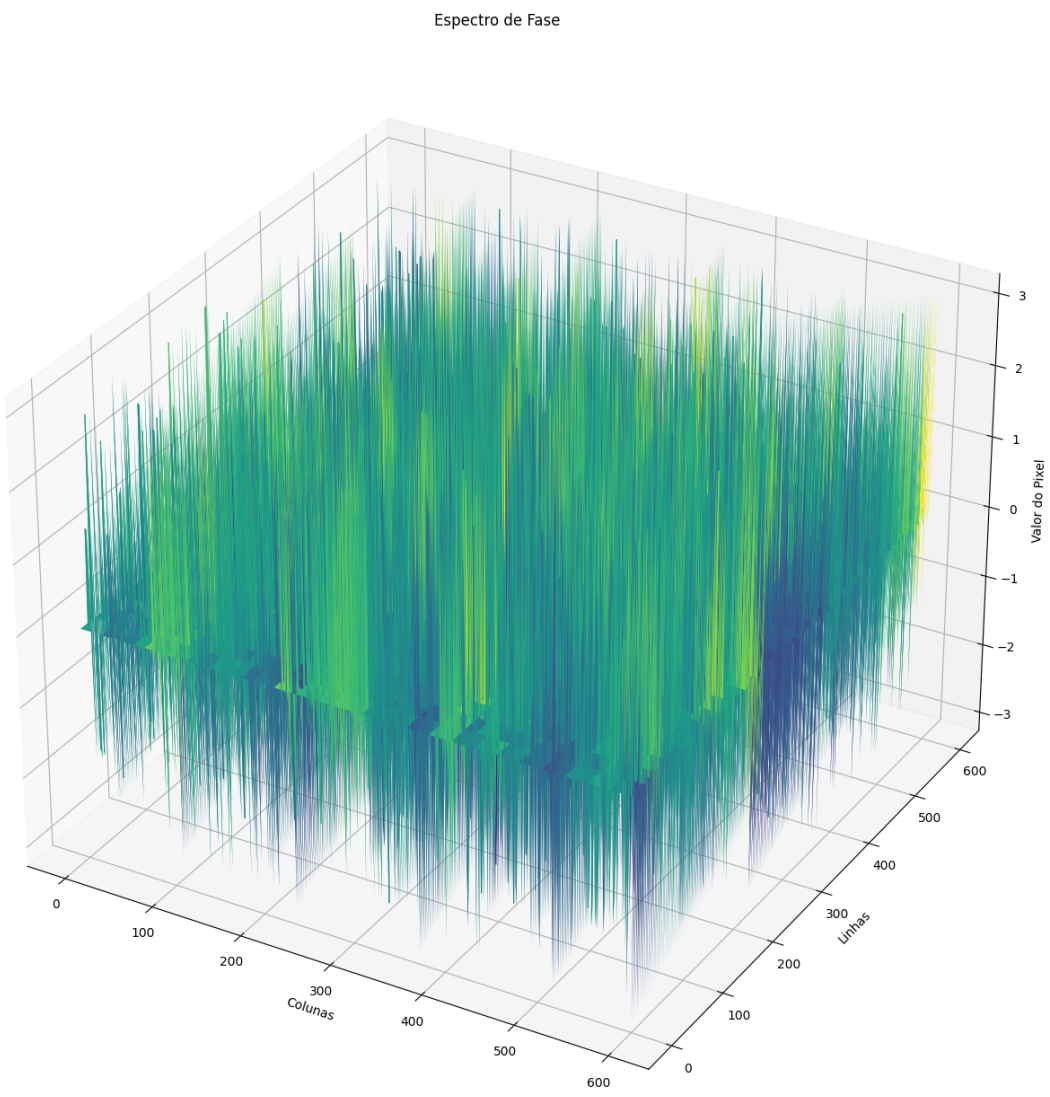
[156]:
```
img_white_square = np.zeros((600, 600, 3), dtype=np.uint8)
img_white_square[225:375, 225:375] = (255, 255, 255)
# mostrar imagem
plt.imshow(img_white_square)
plt.show()

# aplicar a transformada de Fourier
magnitude_spectrum, phase_spectrum, dft_shift =␣
 ↪apply_fourier_transform(img_white_square, display=True)

# aplicar a transformada inversa de Fourier
inverse_fourier_img = apply_inverse_fourier_transform(img_white_square,␣
 ↪dft_shift)
```




```
<Figure size 640x480 with 0 Axes>
```




Espectro de Magnitude
Colunas
Linhas
Valor do Pixel
0
100
200
300
400
500
600
0
50
100
150
200
250



[ ]:
```
#%cd /content/drive/MyDrive/Colab Notebooks/aula7
#! sudo apt update
#! sudo apt-get install texlive-full
! jupyter nbconvert --to pdf TransformadaDeFourier.ipynb
```