**PROGRAM:**

**import tensorflow as tf**

**import tensorflow\_addons as tfa**

**import matplotlib.pyplot as plt**

**# Define the generator and discriminator networks**

**generator = tfa.networks.pix2pix.pix2pix\_generator**

**discriminator = tfa.networks.pix2pix.pix2pix\_discriminator**

**# Load the Cycle-GAN model**

**model = tfa.networks.CycleGAN(**

**generator\_fn=generator, discriminator\_fn=discriminator,**

**lambda\_cycle=10.0, lambda\_identity=0.5**

**)**

**# Load pre-trained weights**

**model.load\_weights("cycle\_gan\_weights.h5")**

**# Function to transform input image**

**def transform\_image(image):**

**# Preprocess image (normalize, resize, etc.)**

**image = preprocess\_image(image)**

**# Perform image transformation using Cycle-GAN**

**transformed\_image = model.predict(image)**

**# Post-process transformed image (denormalize, resize, etc.)**

**transformed\_image = postprocess\_image(transformed\_image)**

**return transformed\_image**

**# Function to preprocess input image**

**def preprocess\_image(image):**

**# Implement image preprocessing steps (e.g., normalization, resizing)**

**return image**

**# Function to postprocess transformed image**

**def postprocess\_image(image):**

**# Implement postprocessing steps (e.g., denormalization, resizing)**

**return image**

**# Example usage**

**input\_image = tf.random.normal([1, 256, 256, 3]) # Example input image**

**output\_image = transform\_image(input\_image)**

**# Visualize input and output images**

**plt.subplot(1, 2, 1)**

**plt.title('Input Image')**

**plt.imshow(input\_image[0].numpy())**

**plt.axis('off')**

**plt.subplot(1, 2, 2)**

**plt.title('Transformed Image')**

**plt.imshow(output\_image[0].numpy())**

**plt.axis('off')**

**plt.show()**