**Cod\_Clock\_coment**

# Importa a biblioteca OpenCV para manipulação de imagens e visão computacional

import cv2

# Importa função específica do Google Colab para exibir imagens no notebook

from google.colab.patches import cv2\_imshow

# Carrega o classificador Haar Cascade treinado para detectar relógios a partir de um arquivo XML

detec\_clock = cv2.CascadeClassifier('/content/drive/MyDrive/Visão Computacional Guia Completo/Cascades/clocks.xml')

# Lê a imagem do relógio a partir do caminho especificado

img = cv2.imread('/content/drive/MyDrive/Visão Computacional Guia Completo/Images/clock.jpg')

# Verifica se a imagem foi carregada corretamente

if img is None:

    # Caso não tenha sido carregada, exibe mensagem de erro

    print('Img not found')

else:

    # Converte a imagem para tons de cinza (recomendado para detecção com Haar Cascades)

    gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

    # Aplica o classificador Haar Cascade para detectar objetos (relógios) na imagem em tons de cinza

    # scaleFactor: quanto a imagem é reduzida a cada escala (menor = mais precisa, mas mais lenta)

    # minNeighbors: quanto maior, menos falsos positivos, mas pode perder objetos reais

    clock = detec\_clock.detectMultiScale(gray, scaleFactor=1.03, minNeighbors=1)  # minSize pode ser usado para filtrar por tamanho mínimo

    # Exibe a quantidade total de relógios detectados

    print(f"Tot Clock: {len(clock)}")

    # Exibe as coordenadas (x, y, largura, altura) de cada relógio detectado

    print("Coordenadas the clock: ", clock)

    # Para cada relógio detectado, desenha um retângulo verde ao redor dele

    for (x, y, w, h) in clock:

        cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)

    # Apenas repete o total de relógios (essa linha não é necessária, já foi impresso acima)

    len(clock)

    print(f"Tot Clock: {len(clock)}")

    # Exibe a imagem final com os retângulos desenhados ao redor dos relógios detectados

    cv2\_imshow(img)