Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Santa Helena
Bacharelado em Ciência da Computação

Prof.: Anderson Brilhador
Aluno: Nathan Guimarães

Data: 03/04/2025
RA: 2153319

Link para código Visão Computacional - Exercícios de Fixação .ipynb
Link para o repositório no Github: visao-computacional

Questão 1: Baixe a imagem vermelho.jpg no moodle, e modifique as cores do pixels da imagem para a cor preta e salve como preta.jpg

Código:

```python
import cv2
import numpy as np

img = cv2.imread('vermelho.jpg')          # Carrega a imagem original
img_preta = np.zeros_like(img)            # Cria uma matriz de pixels
pretos
cv2.imwrite('preta.jpg', img_preta)       # Salva a imagem resultante
```

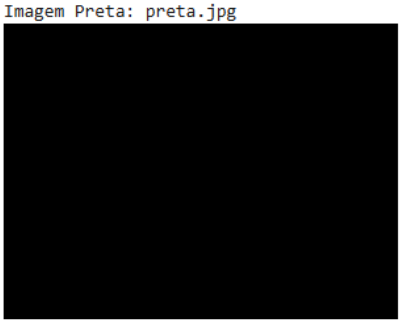
Saída:

```
===== Exercícios de fixação =====

Por favor, faça upload da imagem 'vermelho.jpg' para o Colab
Escolher arquivos vermelho.jpg
• vermelho.jpg(image/jpeg) - 2372 bytes, last modified: 03/04/2025 - 100% done
Saving vermelho.jpg to vermelho (2).jpg

=== PROCESSANDO QUESTÃO 1 ===
✓ preta.jpg criada com sucesso.

Imagem Preta: preta.jpg
```



Questão 2: Abra a imagem preta.jpg e desenhe um quadrado branco (50x50 pixels) no centro da imagem e salve como quadradobranco50.jpg

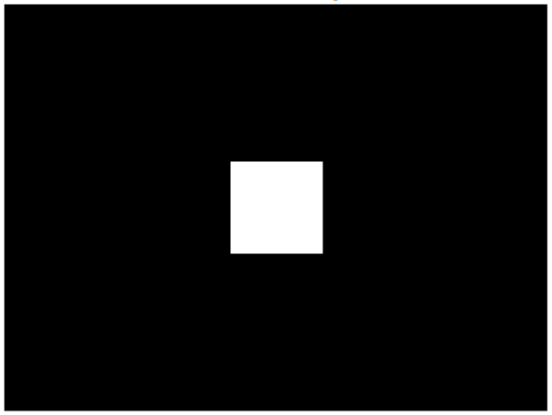
Código:

```python
altura, largura, _ = img.shape
centro_x, centro_y = largura // 2, altura // 2  # Calcula o centro
img_quad50 = img_preta.copy()
cv2.rectangle(img_quad50,
              (centro_x - 25, centro_y - 25),   # Canto superior esquerdo
              (centro_x + 25, centro_y + 25),   # Canto inferior direito
              (255, 255, 255), -1)             # Cor branca e preenchimento
cv2.imwrite('quadradobranco50.jpg', img_quad50)
```

Saída:

```
=== PROCESSANDO QUESTÃO 2 ===
✓ quadradobranco50.jpg criada com sucesso.

Quadrado Branco 50x50: quadradobranco50.jpg
```



Questão 3: Abra a imagem preta.jpg e desenhe um quadrado branco (25x25 pixels) no centro da imagem e salve como quadradobranco25.jpg.

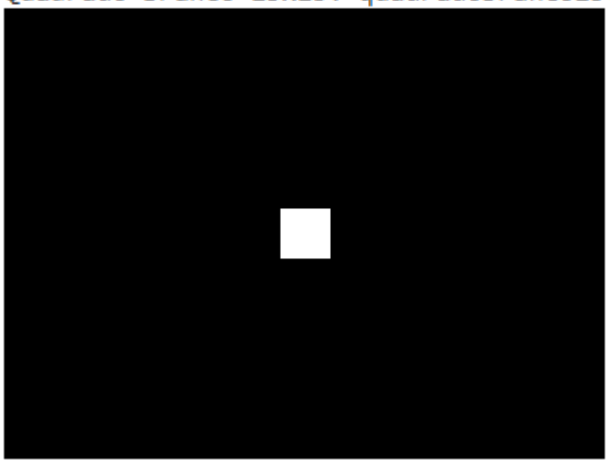
Código:

```python
img_quad25 = img_preta.copy()
cv2.rectangle(img_quad25,
              (centro_x - 12, centro_y - 12),  # Ajuste para 25x25
              (centro_x + 12, centro_y + 12),
              (255, 255, 255), -1)
cv2.imwrite('quadradobranco25.jpg', img_quad25)
```

Saída:

```
=== PROCESSANDO QUESTAO 3 ===
✓ quadradobranco25.jpg criada com sucesso.

Quadrado Branco 25x25: quadradobranco25.jpg
```



Questão 4: Subtraia a imagem quadradobranco25.jpg de quadradobranco50.jpg e salve o resultado como subtracao.jpg.

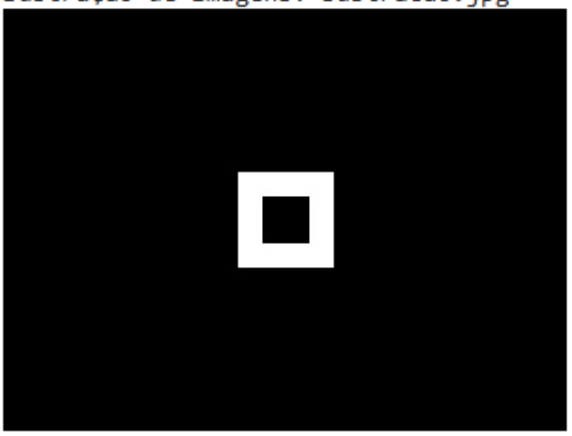
Código:

```python
subtracao = cv2.subtract(img_quad50, img_quad25)  # Subtração pixel a pixel
cv2.imwrite('subtracao.jpg', subtracao)
```

Saída:

```
=== PROCESSANDO QUESTÃO 4 ===
✓ subtracao.jpg criada com sucesso.
```

Subtração de Imagens: subtracao.jpg



Questão 5: Pegue a imagem subtracao.jpg, inverta suas cores, e aplique um XOR com a imagem quadradobranco50.jpg.

Código:

```python
subtracao_invertida = cv2.bitwise_not(subtracao)  # Inverte preto/branco
resultado_xor = cv2.bitwise_xor(subtracao_invertida, img_quad50)  # Aplica XOR
cv2.imwrite('resultado_final.jpg', resultado_xor)
```

Saída:

```
=== PROCESSANDO QUESTÃO 5 ===
✓ resultado_final.jpg criada com sucesso.

Resultado Final (XOR): resultado_final.jpg
```

Função principal :

```
# Executanbdo as questões
def main():
    print("===== Exercícios de fixação =====")
    print("\nPor favor, faça upload da imagem 'vermelho.jpg' para o Colab")
    files.upload()

    # Verificar se o upload foi feito corretamente
    if not os.path.exists('vermelho.jpg'):
        print("\n⚠ ATENÇÃO: Você precisa fazer upload do arquivo 'vermelho.jpg' primeiro!")
        print("Por favor, execute esta célula novamente após o upload.")
        return

    # Executar cada questão
    img_preta = questao1()
    img_quad50 = questao2(img_preta)
    img_quad25 = questao3(img_preta)
    img_subtracao = questao4()
    img_resultado = questao5()

    # Oferecer opção de baixar todas as imagens
    download_all_images()

    print("\n Processamento concluído com sucesso!")

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

Para elaboração das questões 6 e 7 preparei outra célula no mesmo notebook do google colab.

6. Baixe e abra a imagem **televisao.jpg** e recorte a televisão presente na imagem removendo sua base. Depois inverta a imagem horizontalmente. Dica: a televisão tem cerca de 190 pixels de altura e 270 pixels de largura.

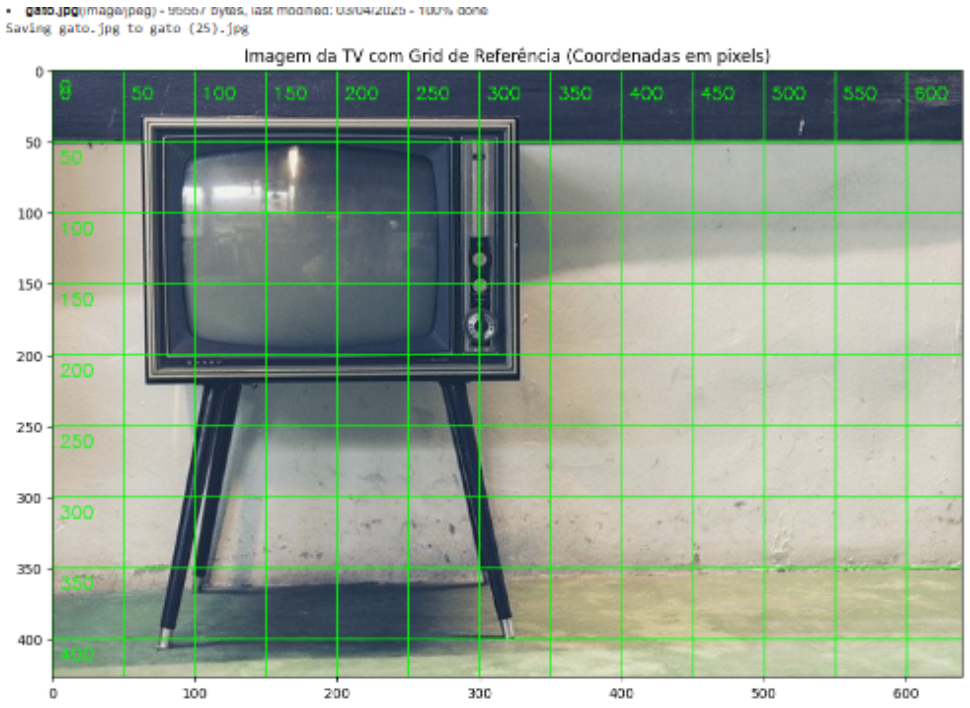
7. Baixe e redimensione a imagem **gato.jpg** para 190 pixels de altura e 268 pixels de largura e misture com a imagem da televisão recortada de forma que o gato apareça dentro da tela da tv. Dica: crie uma máscara retangular com o tamanho da tela (Observação a máscara pode ser criada usando a biblioteca numpy).

Uma **máscara de imagem** é um array binário (geralmente com valores 0 e 1, ou 0 e 255) usado para selecionar regiões específicas de uma imagem. Essa máscara serve como um filtro lógico que define quais pixels devem ser mantidos ou removidos em uma operação sobre a imagem.

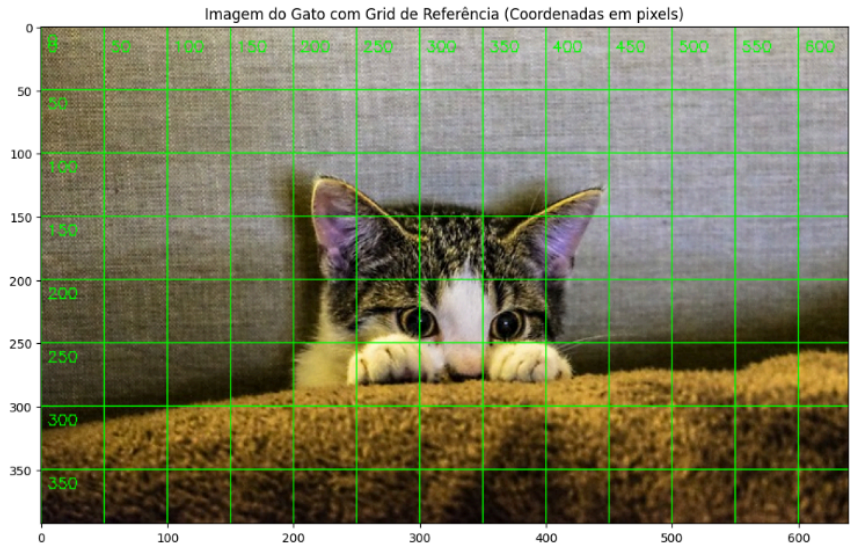
 # aplicando a operação and entre a imagem do gato e a mascara
imagem_gato_mascara = cv2.bitwise_and(imagem_redimensionada, imagem_redimensionada, mask=mascara)

 # misturando as imagens
imagem_resultante = cv2.addWeighted(tv_flip_horizontal, 0.5, imagem_gato_mascara, 0.5, 0)

Saída:




```
Dimensões da imagem da TV: 640x427 pixels
Observando o grid acima, digite as coordenadas para recorte:
Coordenada X inicial (sugestão: 185): 60
Coordenada Y inicial (sugestão: 98): 20
Coordenada X final (sugestão: 330): 330
Coordenada Y final (sugestão: 210): 225
```




```
Verificação final de dimensões:
TV: 270x205
Gato: 270x205
```


```
Imagem resultante salva como gato_na_tv_final.jpg
Deseja baixar a imagem resultante? (s/n): s
```

- Durante o processo foi encontrado erros ao mesclar as imagens devido a proporção das dimensões serem diferentes, foi necessário um tratamento de modo que as duas imagens após o recorte fossem de tamanhos iguais, foi realizada a tentativa de modelar um retângulo nas proporções corretas da tv, no entanto, novamente na tentativa de mesclar as imagens devido a inversão que estava ocorrendo depois de configurar as dimensoes do retangulo, ocorreu um erro de incompatibilidade sugerindo mais tratamentos para redimensionar a imagem do gato. A solução foi recorrer a uma proporção que ao ser invertida fosse próxima as dimensões definidas no recorte da imagem do gato que ao ser mesclada toma as formas aproximadas da tela da TV.