CVI620 – Assignment 1

Summer 2025

Total Mark:	7.5 marks (7.5% of the total course grade)
Submission file(s):	- Assignment1.py or Assignment1.ipynb (you can upload multiple files) - Assignment1.docx (this document with your answers)
Deadline	- May 28th, 2025

If you are unable to complete the assignment on-time for any legit reason, please provide documentation explaining your absence (e.g., an appointment confirmation or a work letter).

Please be aware that the assignment is designed to make you **research** and develop.

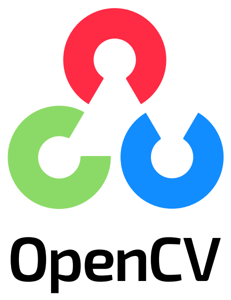
Please submit the submission file(s) through Learn@Seneca. Make sure to use GitHub and provide the link to your GitHub account for all your contributions in the box below:

Project GitHub repository:	

Please paste the resulting images and answers in this document.

Part I: A photo booth application

Recreate the OpenCV logo using OpenCV drawing functions only (e.g., cv2.circle, cv2.line, etc.).



You must draw:

- Three colored shapes (ellipse: Blue, Green, Red) arranged in a triangular pattern.
- Proper positioning and size of circles.
- The text "OpenCV" at the center/bottom.

Part II: Image Arithmetic

Write a Python script to manually blend two images using NumPy operations.

Use the following formula to blend them:

$$\text{blend} = (1-\alpha)\cdot \text{img1} + \alpha\cdot \text{img2}$$

- alpha must be a value between 0 and 1.
- Do not use cv2.addWeighted.
- Display and save the blended image as "manual_blend.jpg"

Part II: Image Arithmetic

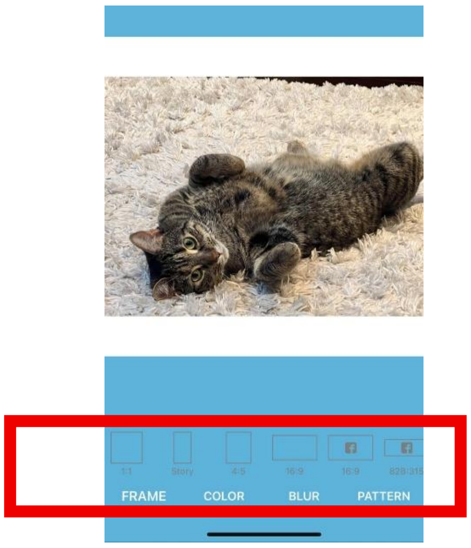
Design a modular photo editing application in Python using OpenCV and NumPy. The app should allow users to load an image and apply a sequence of processing steps interactively through a menu interface.

Required Functionalities:

- Load an image from file.
- Show the following menu:

```
==== Mini Photo Editor ====

1. Adjust Brightness

2. Adjust Contrast

3. Convert to Grayscale

4. Add Padding (choose border type)

5. Apply Thresholding (binary or inverse)

6. Blend with Another Image (manual alpha)

7. Undo Last Operation

8. View History of Operations

9. Save and Exit
```

- Brightness/Contrast
- Padding: Ask the user to specify the padding size and border type (constant, reflect, replicate, etc.). Additionally, include an option for the user to choose the padding proportion: Square, Rectangle, Custom Ratio (e.g., 4:5)



If the user selects a custom ratio like 4:5, your program should calculate and apply the padding so that the final image respects the chosen aspect ratio, regardless of the original size. The user should also be able to adjust the total padding size, and your code must maintain the proportion accordingly. (add smallest padding at the beginning to make it rectangle, then increase or decrease the padding size of the user wants)

- Thresholding: Let user choose between cv2.THRESH_BINARY and cv2.THRESH_BINARY_INV.
- Blending: Ask for a second image path and alpha (0 to 1).
- Undo Feature: Keep a history stack of image states. Allow the user to revert to the previous state.
- History Log: Keep a list of all actions performed (e.g., "brightness +50", "padded 20px with reflect"). Display the list before exiting or when requested.

Additional Requirements:

- Code must be modular. Each operation must be a separate function.
- Allow the user to apply multiple transformations in a row.
- When exiting, ask whether the user wants to save the final image, and under what filename.
- Use matplotlib to show side-by-side [original | preview] after every transformation.
- Include some of the results in this document.

.

Part IV:

Academic integrity Policy

I, ------------ (mention your name), declare that I have read and understood the Academic Integrity Policy.

GOOD LUCK!