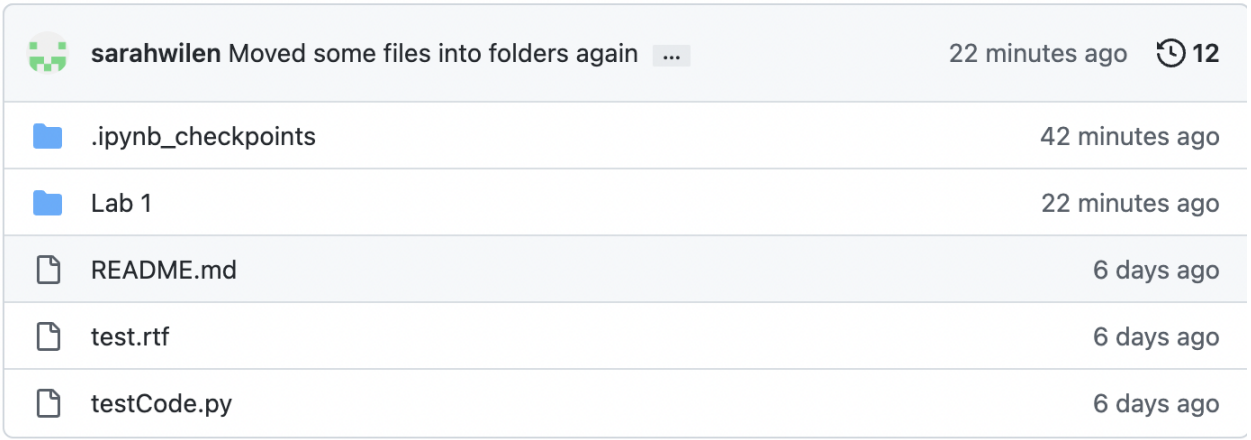
# Lab 1

## Task 1

My github repo: https://github.com/sarahwilen/180DA-WarmUp

## Task 2



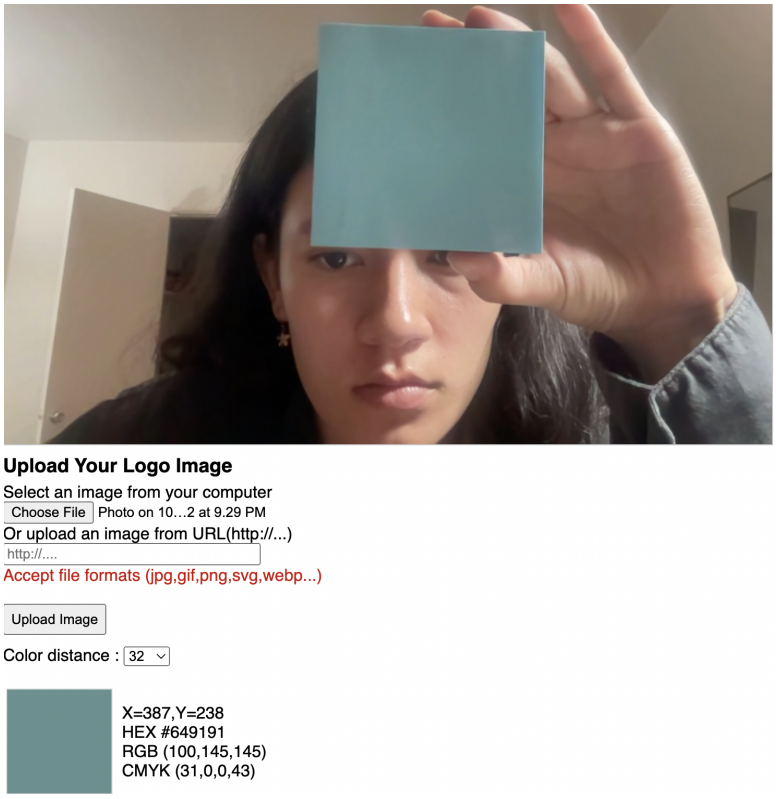

## Task 3

```
(ece180) sarah@MacBook-Pro 180DA-WarmUp % python3 testCode.py
ECE_180_DA_DB - Best class ever
```

## Task 4

### Problem 1

My first step was to determine the estimated RGB values for this blue sticky note. I did this by taking a picture using my laptop video camera and uploading it to a color picker site. I noticed that the sticky note was easier to track in the HSV space, so my first step was to convert my BGR to HSV.




Next, I wrote my script, which I wrote references to in the script itself. Here is a detailed description of how I accomplished the task:

1. I started by turning on my video camera with
    a. `cap = cv.VideoCapture(0)`
2. Then, I started recording using OpenCV's `VideoWriter`
3. While my video camera was still on and active…
    a. I read in each frame of the video
        i. `ret, frame=cap.read()`
    b. I converted each frame from BGR to HSV because I noticed that worked the best for my tracking conditions
        i. `hsv=cv.cvtColor(frame, cv.COLOR_BGR2HSV)`
    c. Next, using the HSV values I found from uploading my picture to a color picker, I created a lower and upper threshold for the blue sticky note
    d. Then, I thresholded the HSV image to find only the section of the image with the blue sticky note
        i. `mask = cv.inRange(hsv,lower_color, upper_color)`
    e. Following, I noticed that since my mask was already in grayscale, I could get the contour directly from that
    f. Starting, I found the canny edges
        i. `edged=cv.Canny(mask,30,200)`
    g. Next, using the canny edges, I found the contours

    i. `contours,hierarchy=cv.findContours(edged,cv.RETR_EXTERNAL,cv.CHAIN_APPROX_NONE)`
    ii. I think since my sticky note was square, I could've used `CHAIN_APPROX_SIMPLE` because it would remove redundant points and compress the contour, saving memory. However, I decided to keep it as `CHAIN_APPROX_NONE` because occasionally, it would make my contour look better because of the bad lighting conditions
  h. Next, I drew my bounding rectangle using each tuple of points in the contour array
  i. Finally, I bitwise-AND the mask and the original image so that I would have a resulting video of me, the sticky note, and the bounding box around the sticky note
  j. I output wrote the videos
4. Using the keyboard key ‘q’, I broke and ended

I noticed that HSV is typically better because HSV distinguishes the image intensity from the color information, which was useful because I performed this lab throughout the day. Besides the complete darkness at night, the lighting change and shadows did not affect my results too much.

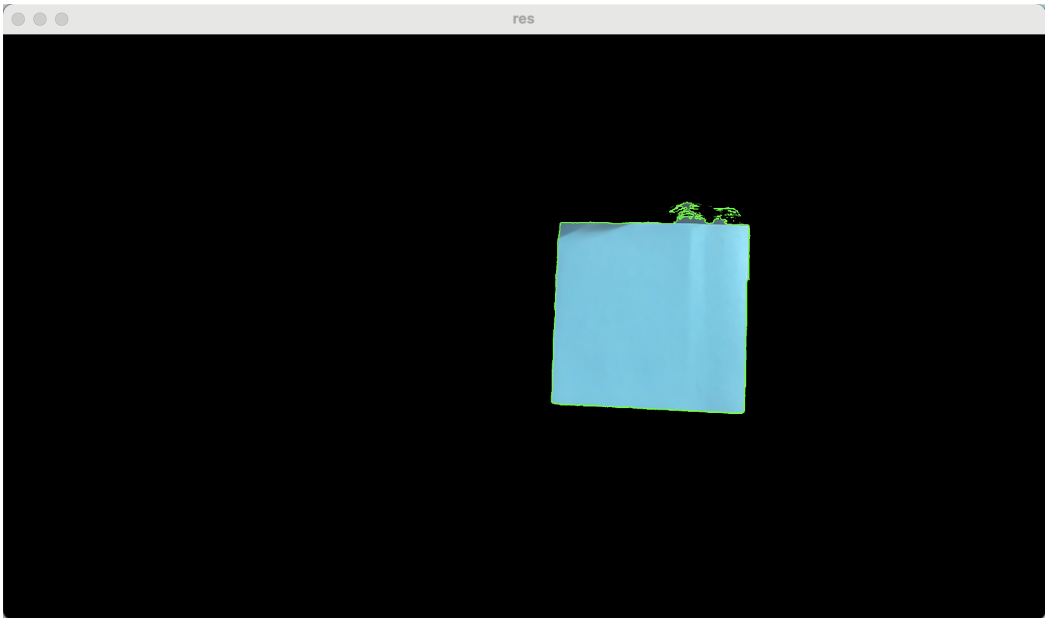
My threshold range for hue was between 80 and 100, for saturation between 50 and 255, and value between 50 and 255. I obtained this threshold from the example problema nd it worked well for my lab.

Difficulties:
By the time I turned in the lab, I still couldn't get the video that was captured to open. At first, I outputted a video file that seemed like it would work, but it was only compatible with Windows, so I couldn’t open it with Mac. Then, I outputted an mp4, which I thought would be compatible with Mac. When I tried to open the file, it just endlessly loaded. I am still going to make changes on the video downloading.

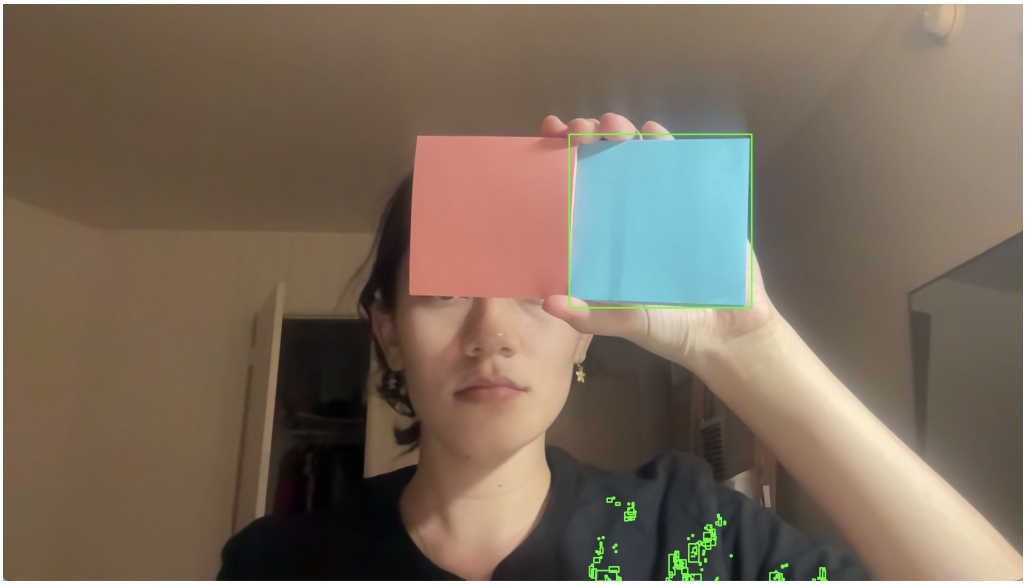
Mask


Canny Edges


Canny Edges + Contouring




Mask + Original Image



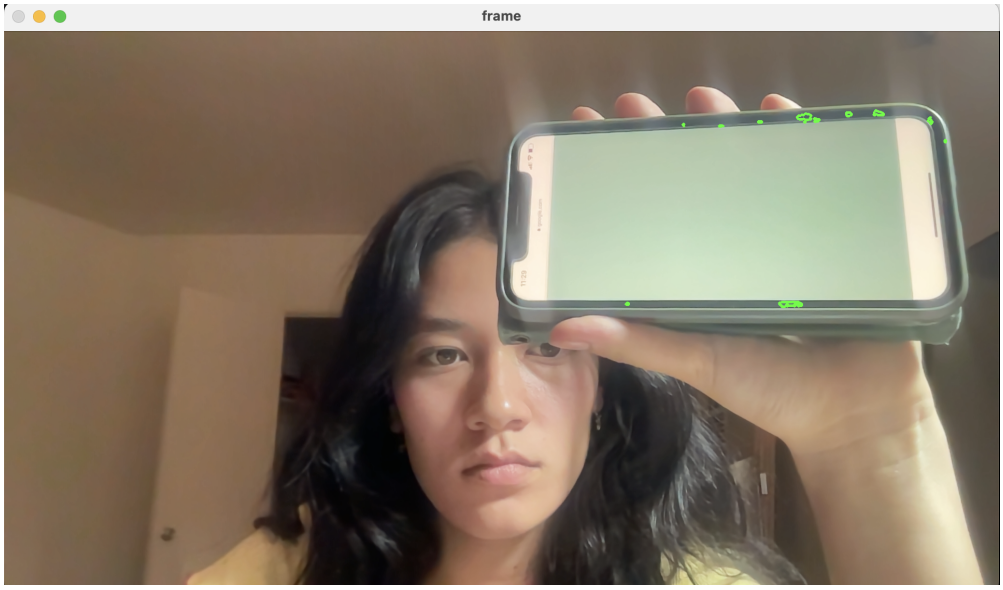
I included a pink post it note next to the blue one to demonstrate the algorithm only tracked the blue sticky note.

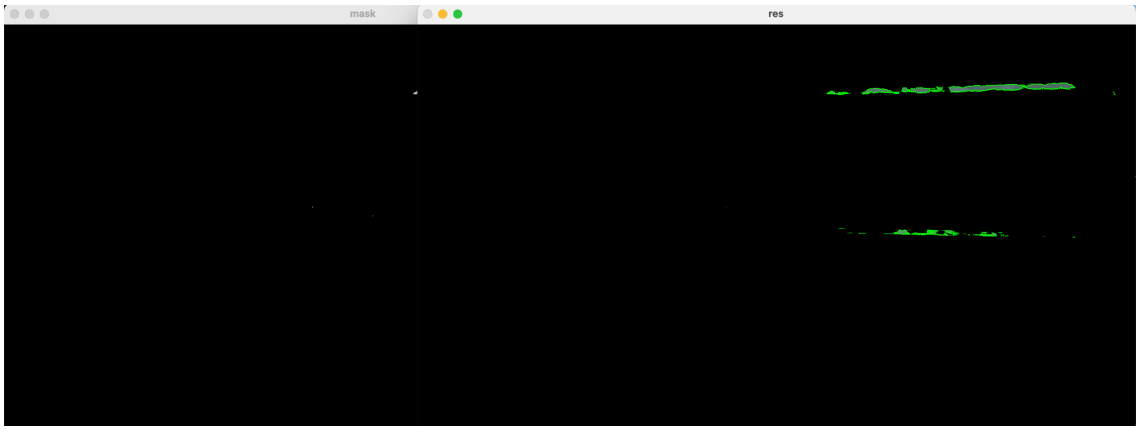
## Problem 2:

At the time that I performed the lab, when I turned the lights off, the room was completely dark, so it could not pick up the color. However, when I shined my phone flashlight on it with low intensity, my object could be picked up, however, if I move the object quickly, it was more difficult to pick up the object.

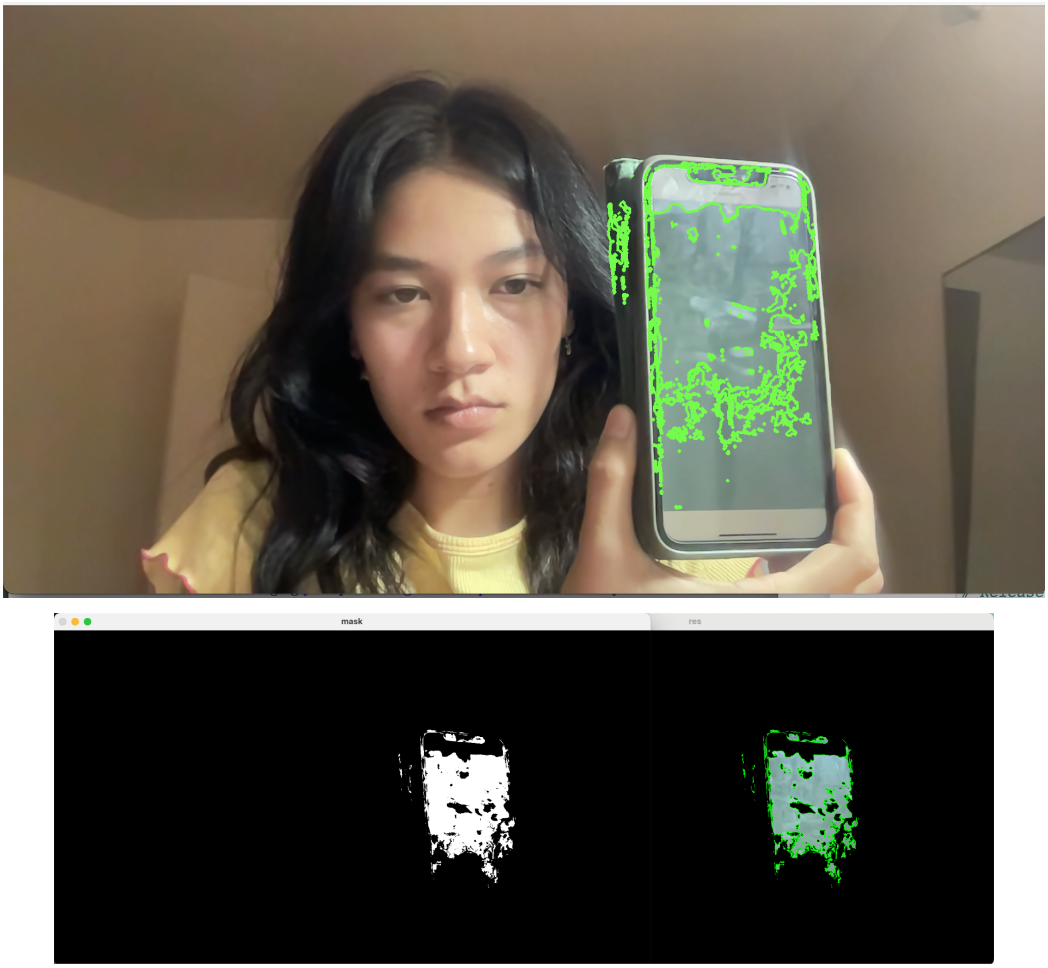
## Problem 3:

Highest brightness:



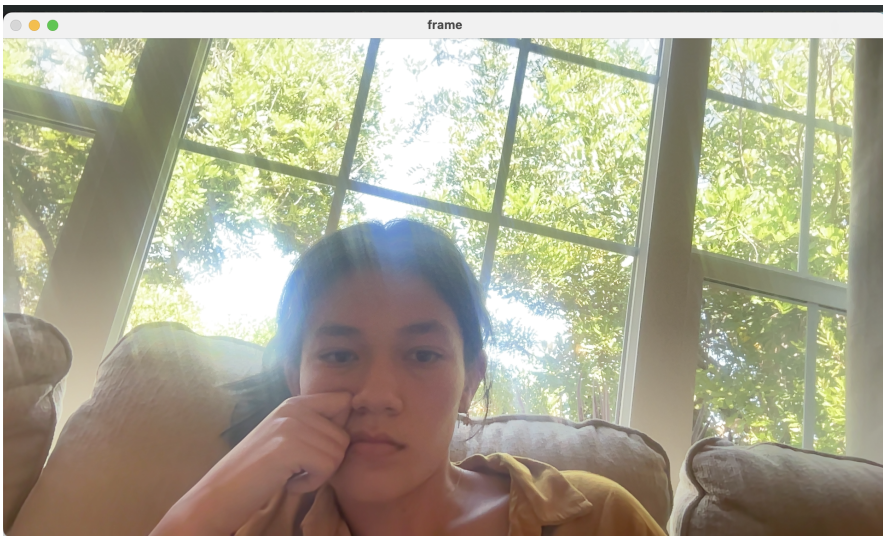




Brightness down:




Changing the phone brightness up hurt how the code was able to track the color a lot more than when I turned the brightness down. I think I might be able to fix this by increasing the threshold HSV values.

# Problem 4:




The non-phone object seemed to be more robust than the phone brightness. I think this is because when the phone brightness is changed, it affects the color more than in a natural environment. I remember a few years ago there was a picture of a dress whose color seemed to change on different screens. I have a feeling that this is the same effect as what is occurring when I change the brightness on my phone screen.

Difficulties:
My issue was that since the k-means was nested inside the frame loop, it was so slow to process the video. When I moved, the video would only register the movement seconds afterwards.