Joshua Austria

180DA Lab 1A

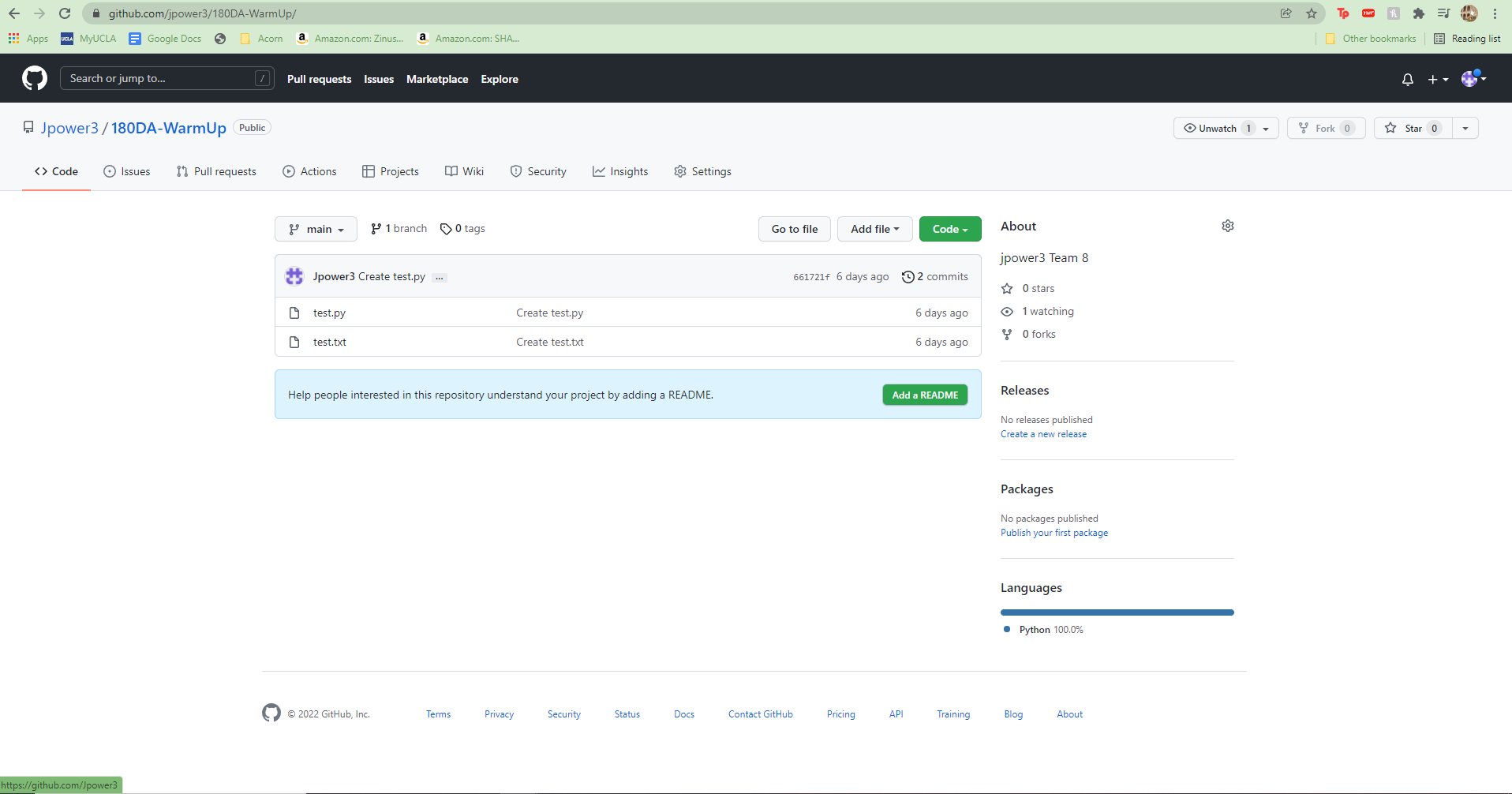
Lab 1

13 January 2022

Introduction to Software Dev and CV

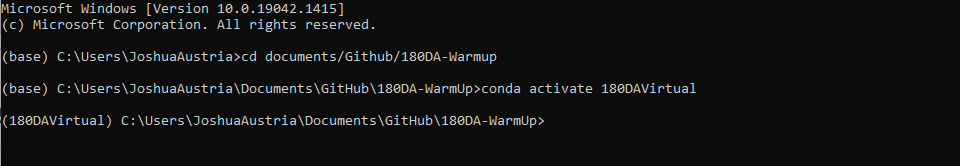
**Task 1:**

Opening up a Github account was simple as I had already created one. From there it was a simple creation of the public repository that is currently available at: www://github.com/jpower3/180DA-WarmUp/

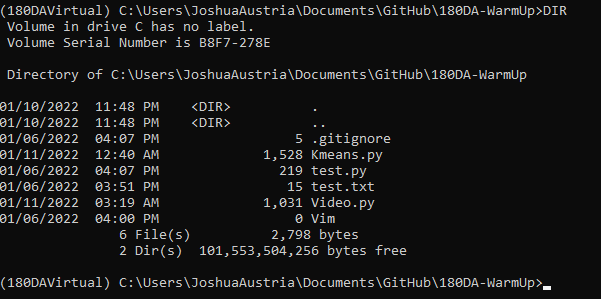


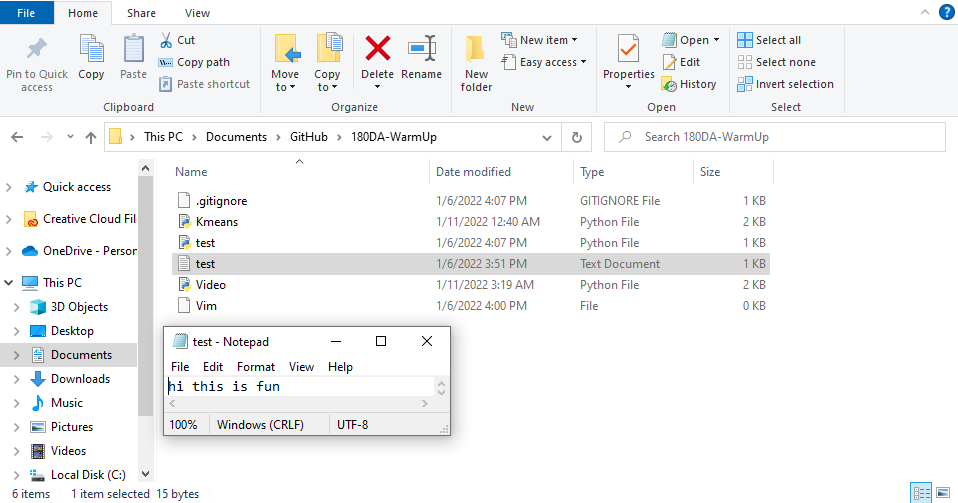
**Task 2:**

I added the virtual environment as defined by section 3. Evidence for that can be found here as I named my virtual environment “180DAVirtual”

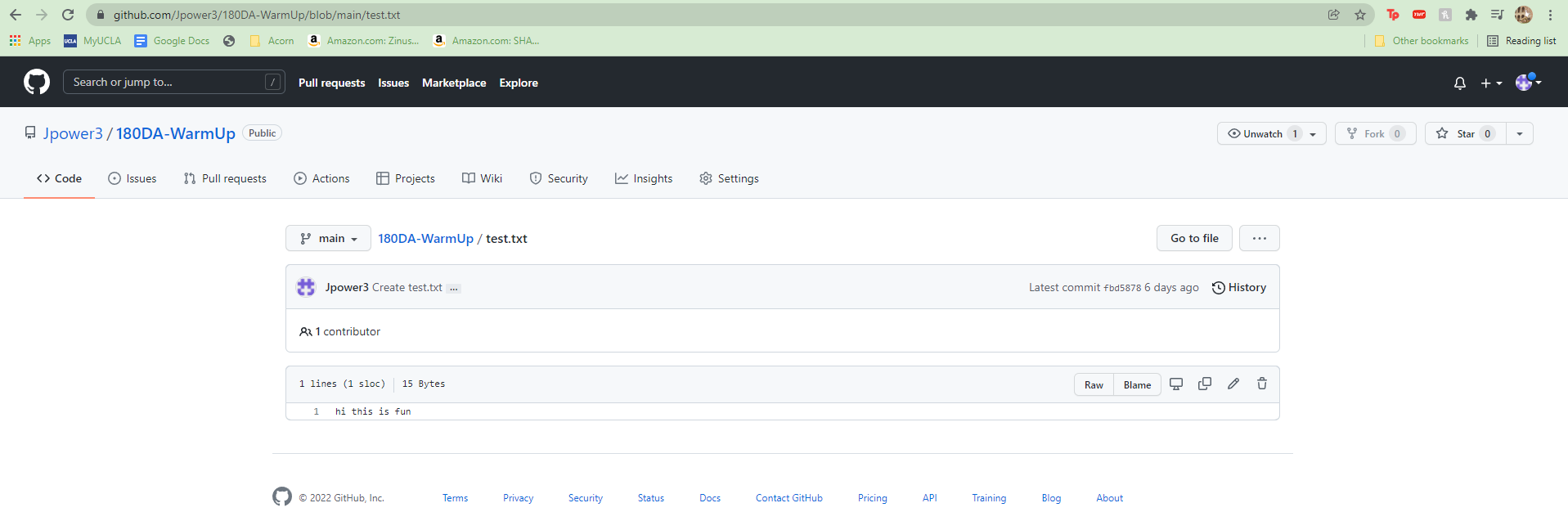


Within this virtual environment I opened up the test.txt file and saved some sample text.



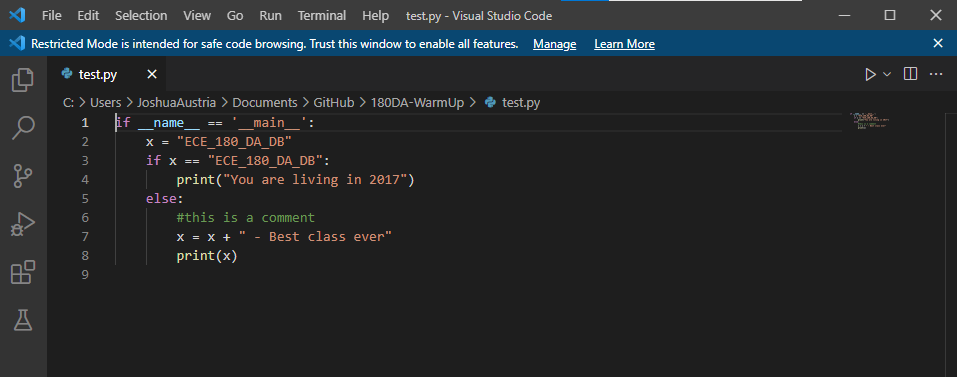


Here is the text that I used to test the virtual environment and GitHub link. I added, committed, and pushed the remote repo utilizing the desktop windows app of GitHub. The text also was added to the GitHub repository correctly.

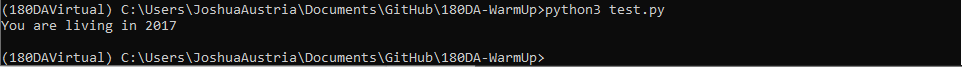


**Task 3:**

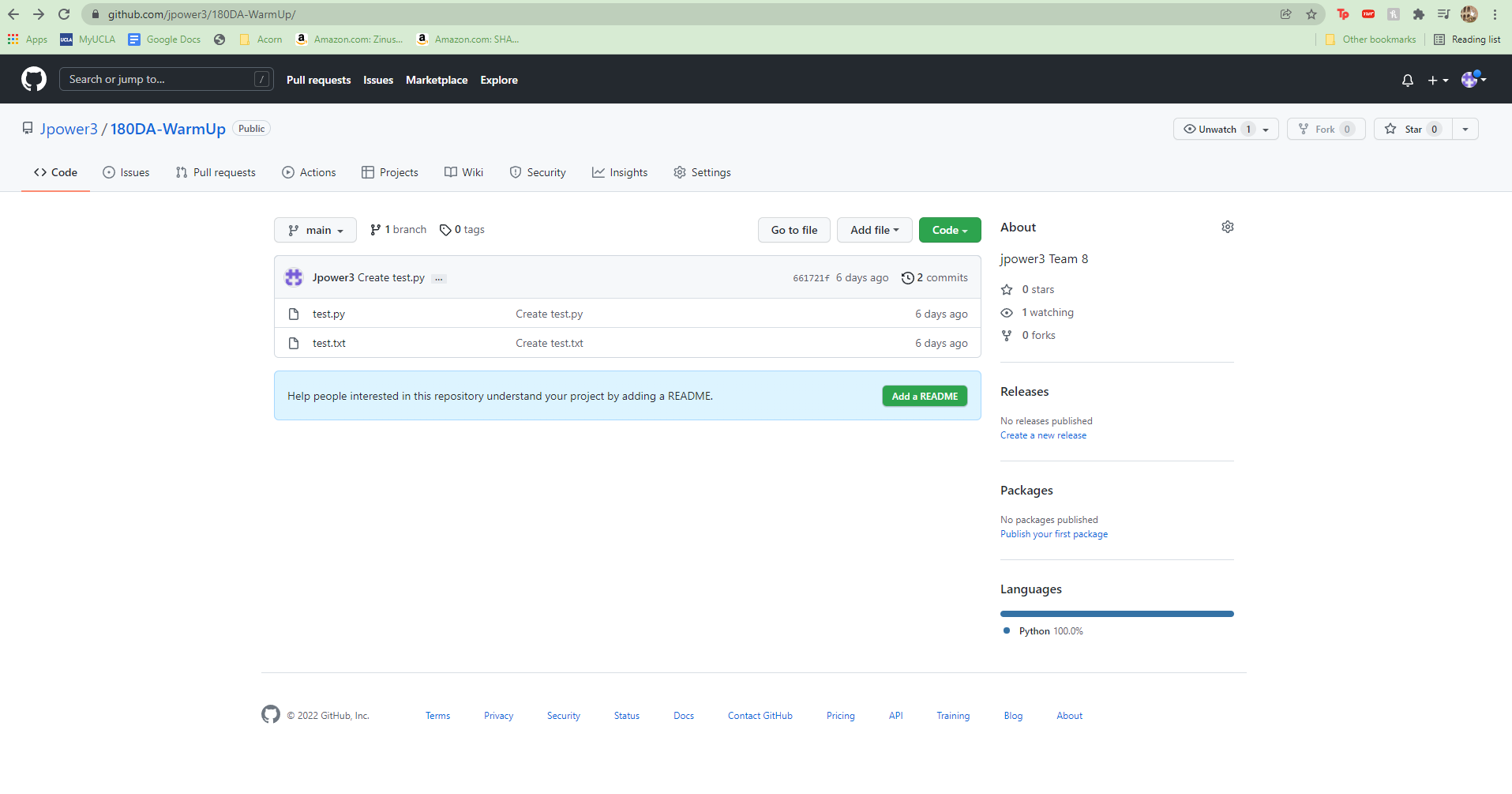
I added the following code utilizing the console and edited it using visual studio code. I will probably try SublimeText at some point as I agree, those colors are very nice.

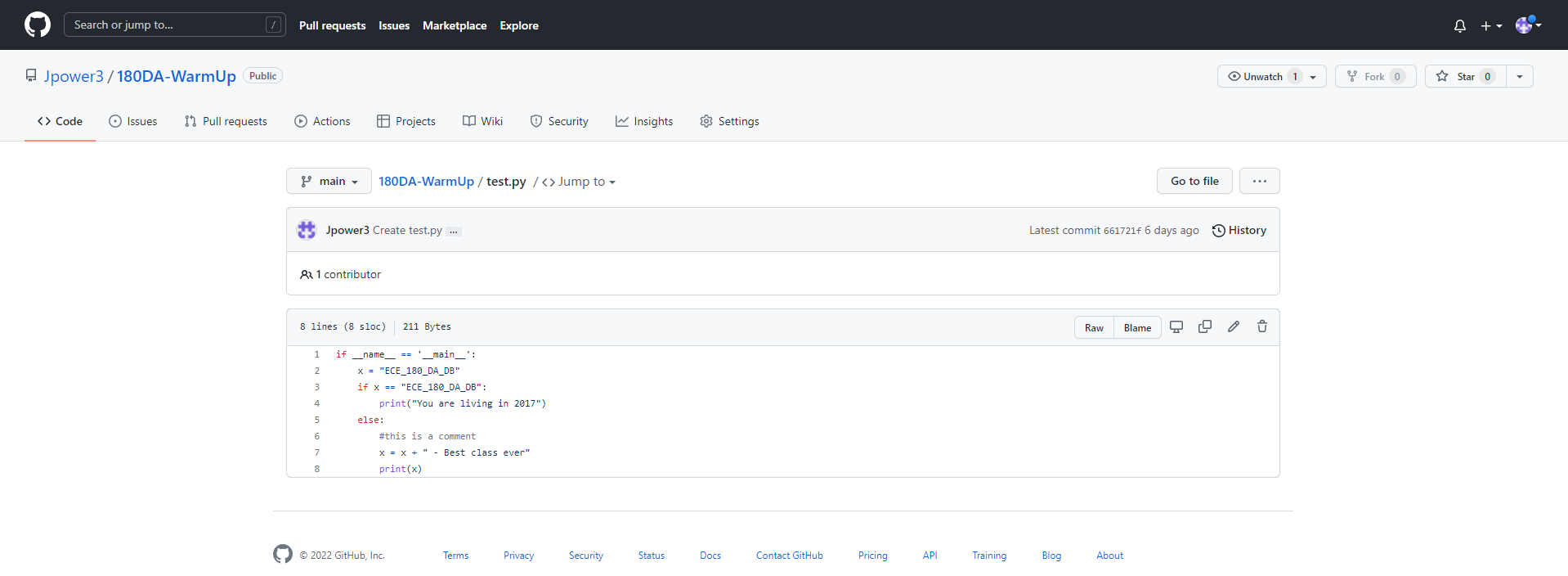


Here is me running the code from the console’s virtual environment.



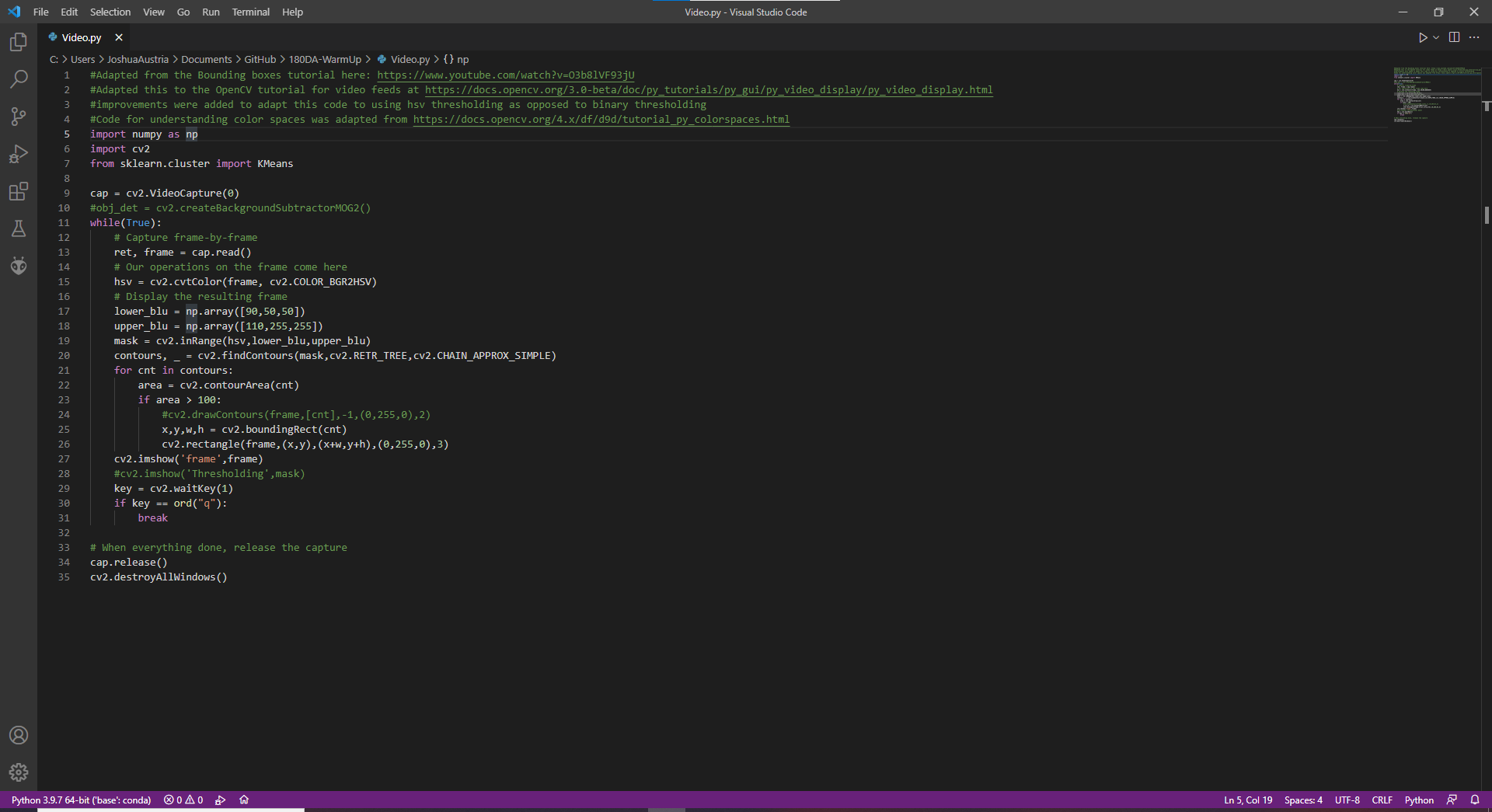
This code was also added, committed, and pushed into my remote repo.

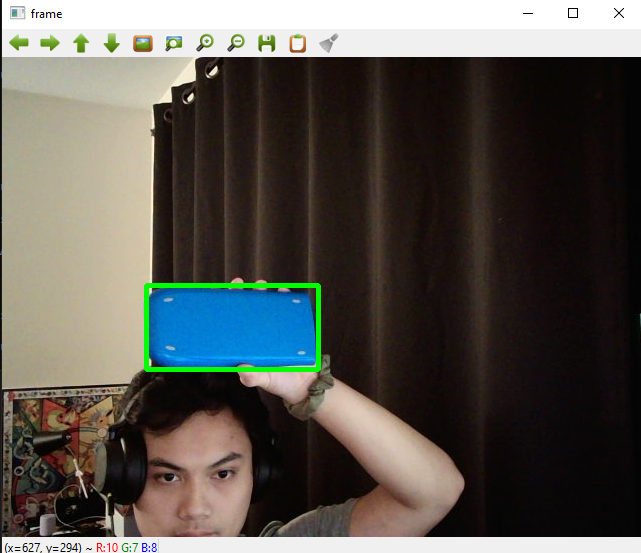


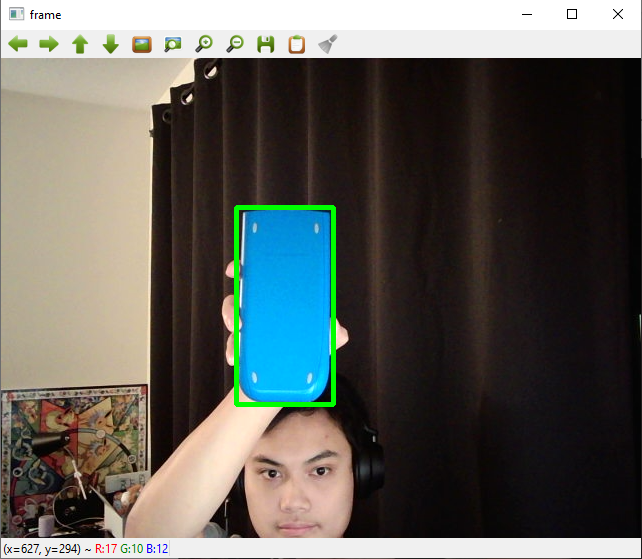


**Task 4:**

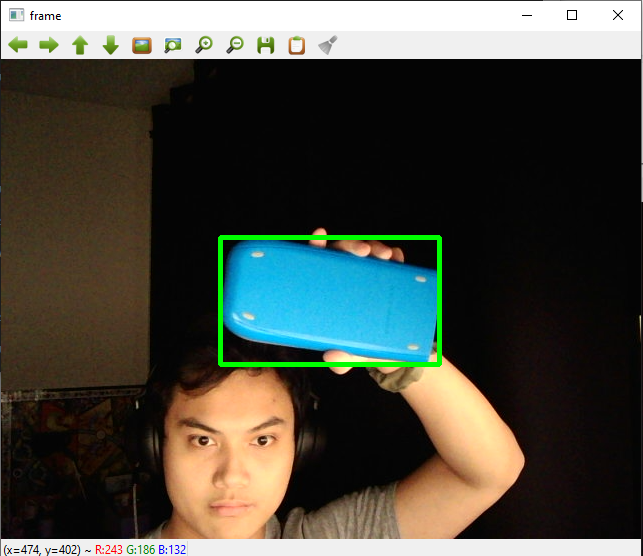
1. I found that HSV is typically easier to track as defining the values for RGB is difficult and inaccurate in its result due to various lighting conditions. Here is a screenshot of me trying to track a calculator with OpenCV. The required threshold was from about 90/179 to 150/179 Hue values with ranges of S and V of 50 to 255 to counteract differences in lighting.

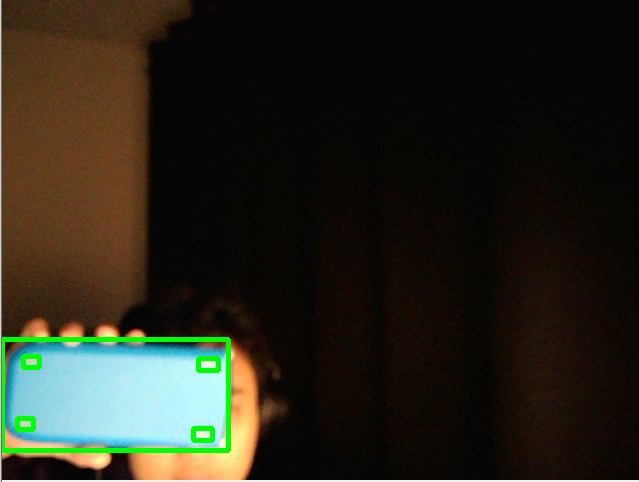




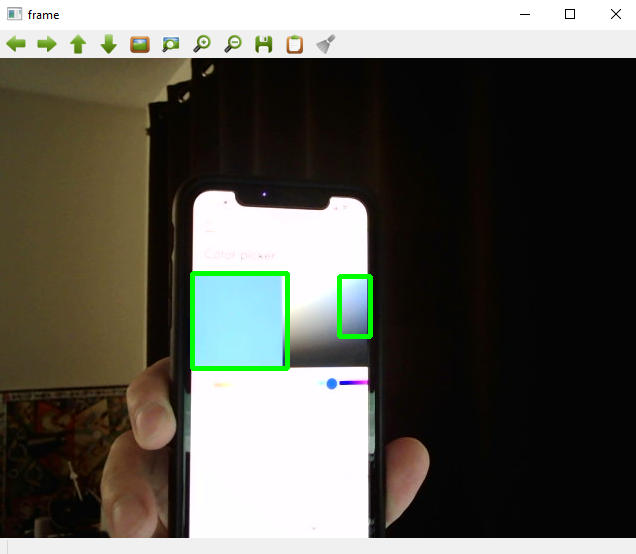


2. By turning off the light, my code was actually able to track the object easier due to the increased contrast between the particular object I chose and the background. Adding light to the scene actually decreases the ability of my code to track the blue colors as blue hues from the light are added to the scene increasing the amount of objects tracked by my code.

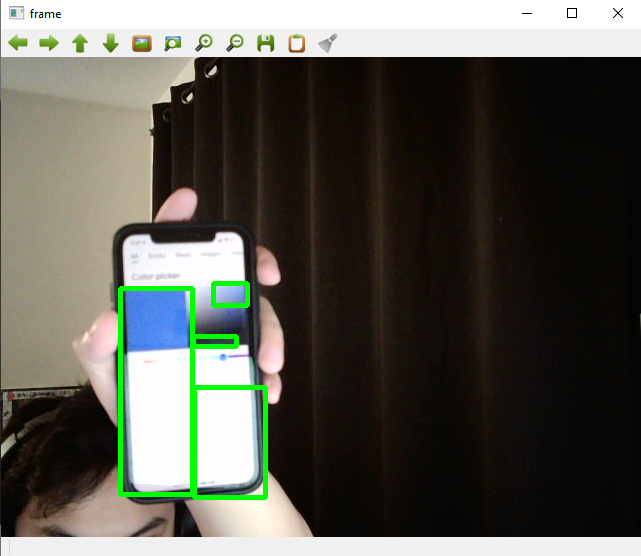




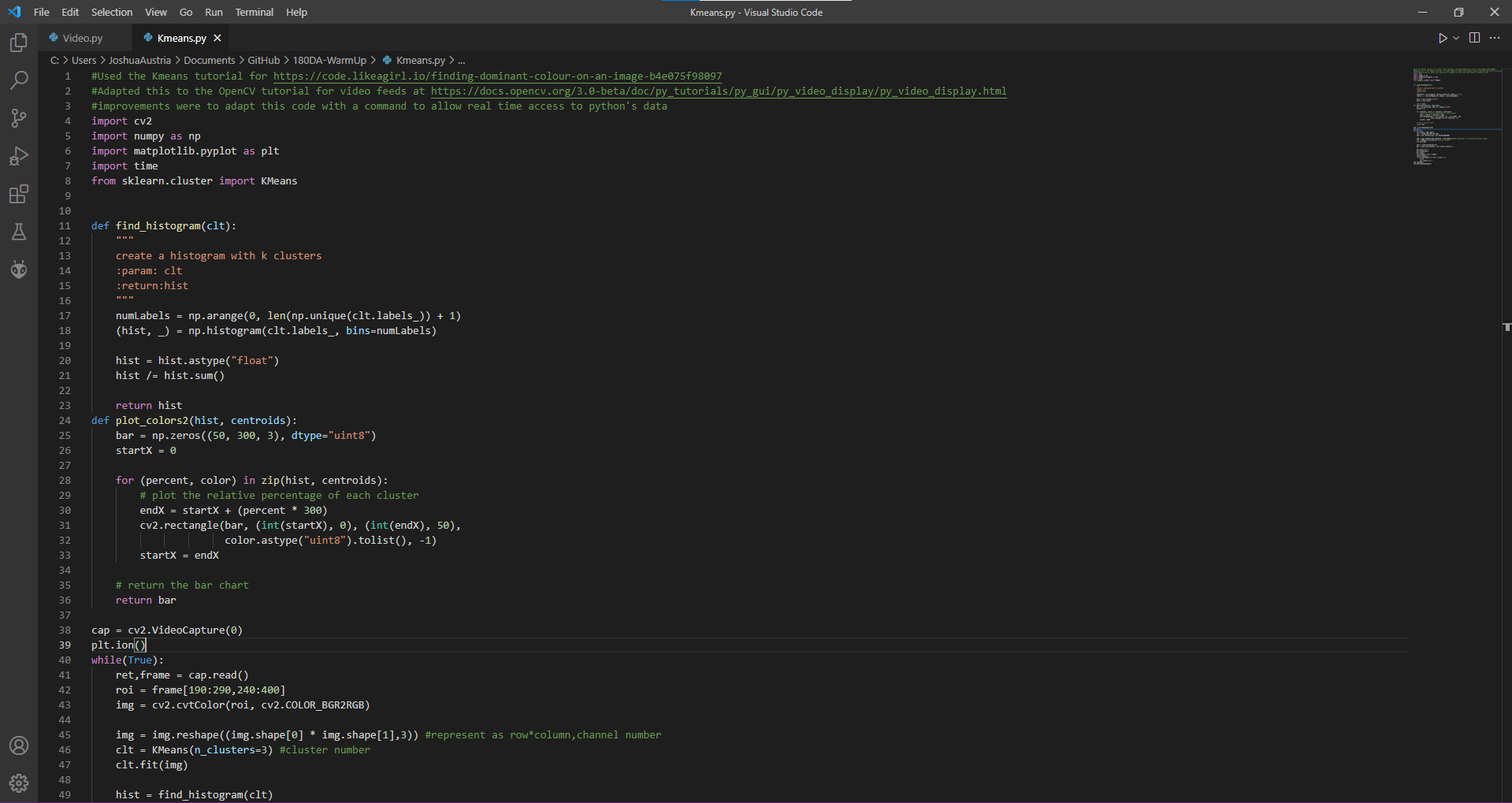
3. Now changing the threshold of the code to 90/179 to 110/179 and using a color picker that places a hue of 100/179 onto my phone, my code is able to accurately track the color with a much smaller range as seen below.

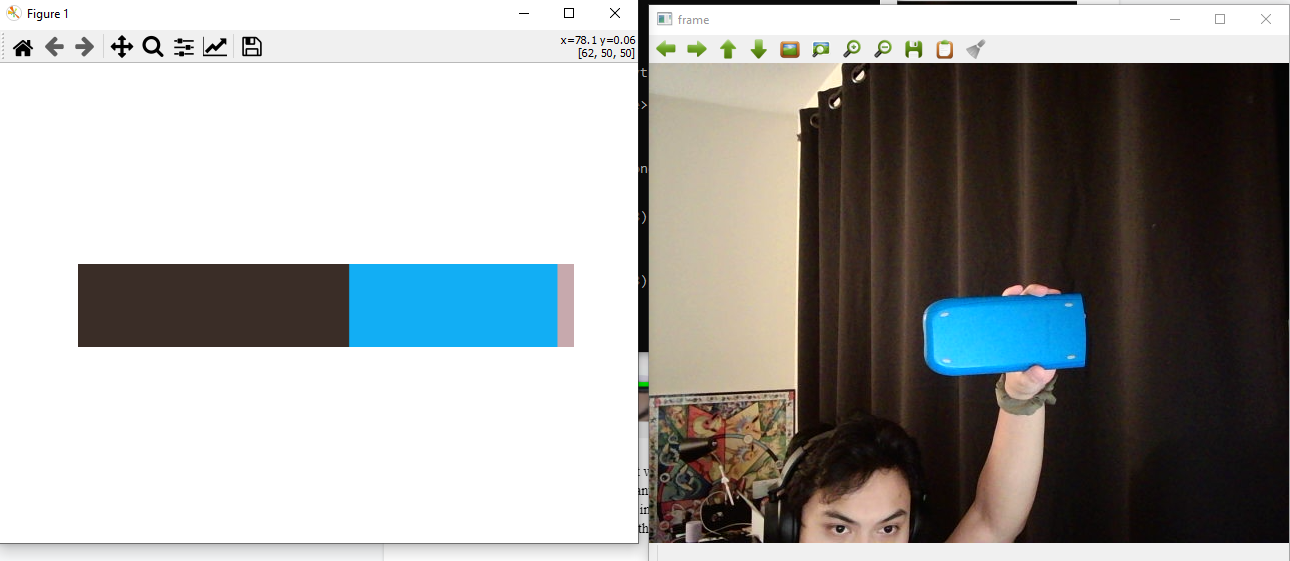


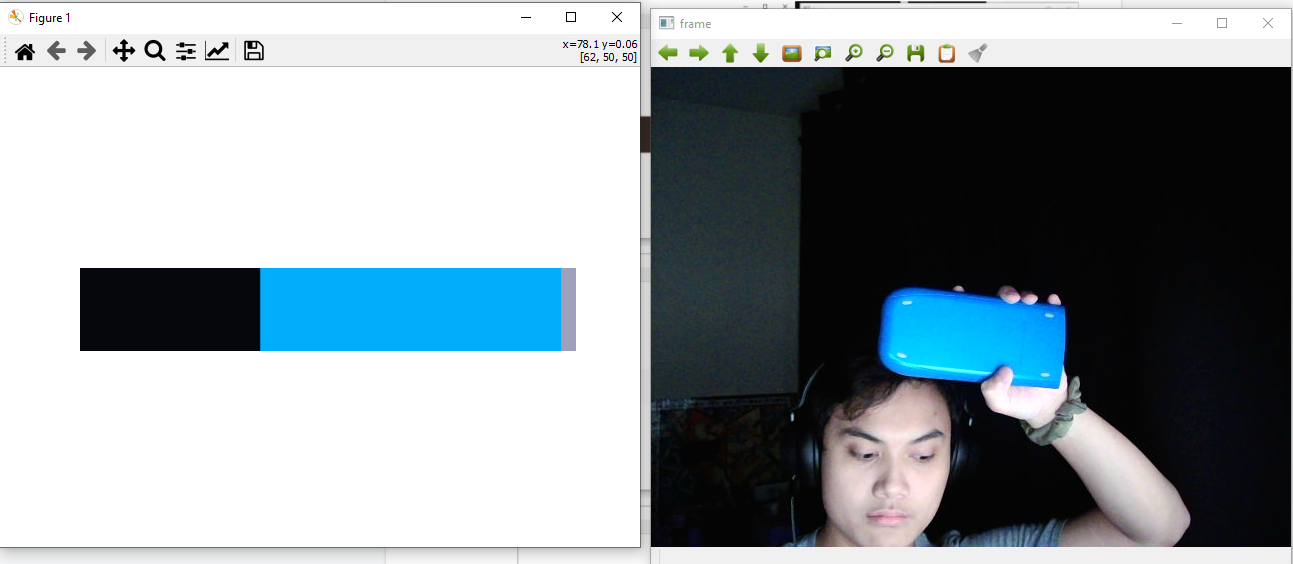
Lowering the brightness of my phone however does exit the threshold of my color picker and cause errors, but this is to be expected as the color threshold takes brightness into account with the HSV model. Other items on the phone screen begin to fall within the threshold barriers of the blue hue and begin to be picked up.

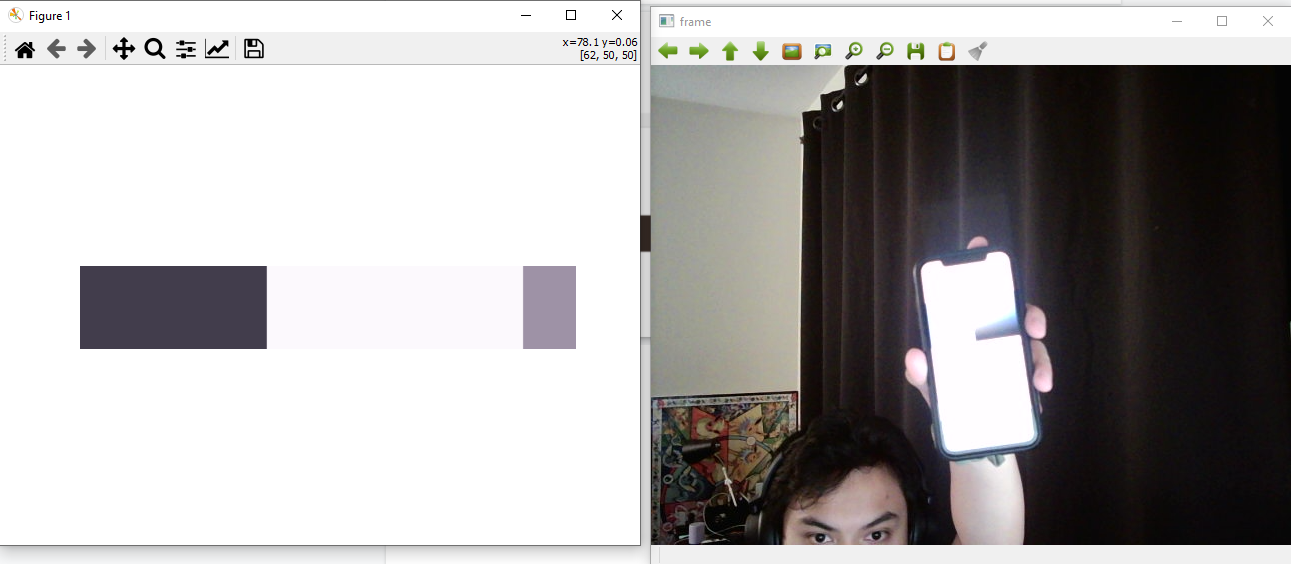


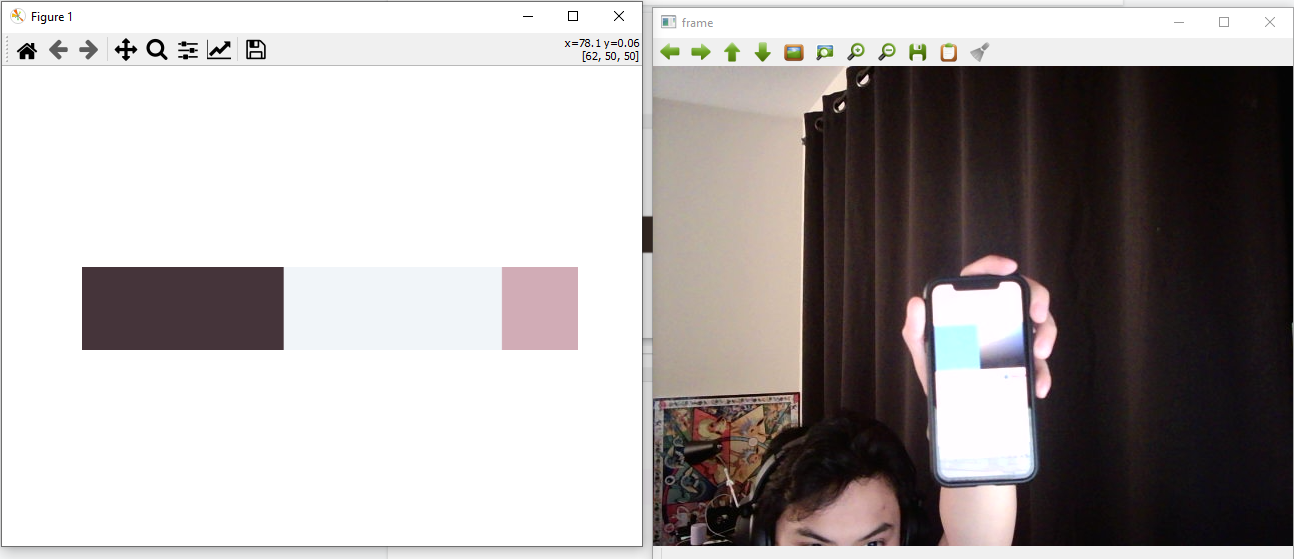
4. I used the code that was sent in collaboration with a K-Means tutorial to create code that tracks the dominant color of a central rectangle in my video feed (to speed up the code). This was done in real time using python’s interactive mode and changes as different things enter the realm of the central rectangle. My phone seemed to be more suscep to changes in its brightness as my calculator maintained its dominant color hue relatively easily despite drastically changing the lighting conditions. The glare from my webcam may have exacerbated the results, but objects that emit their own light seem to do worse than those that are simply colored differently from their surroundings.











All code has been added, committed, and pushed into my public repo.