Work Report

Computer Vision

Suryansh Gupta

28th May, 2019

Summary

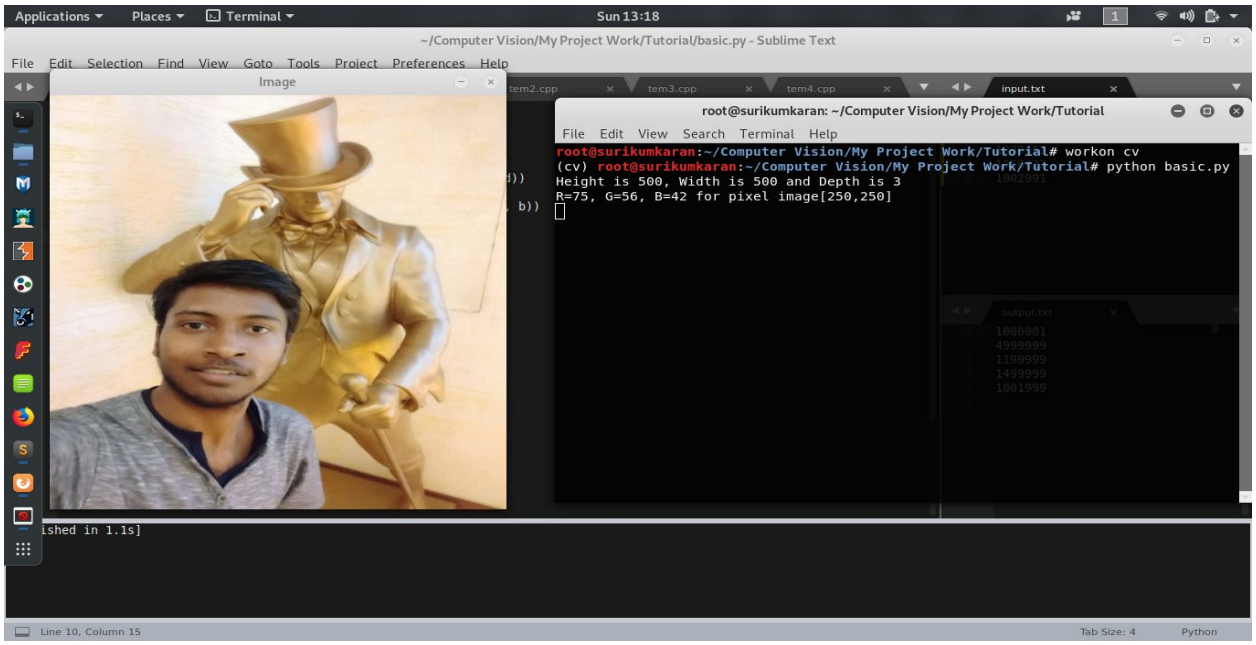
So far I have covered basic of Image Processing using OpenCV and simultaneously taking help from other libraries like imutils, numpy, skimage. The few main learnings are :

- Resizing Images
- Rotating an Image
- Smoothing an Image using GaussianBlur
- Drawing on an Image
- Converting an Image to GrayScale
- Edge Detection
- Thresholding
- Detecting and Drawing Contours
- Erosions and dilations
- Masking and Bitwise AND operation
- Applying perspective transform
- Adaptive Thresholding
- Otsu's Thresholding
- Contour’s Functions
- Building a Document Scanner
- Building a Bubble Sheet Scanner and Grader

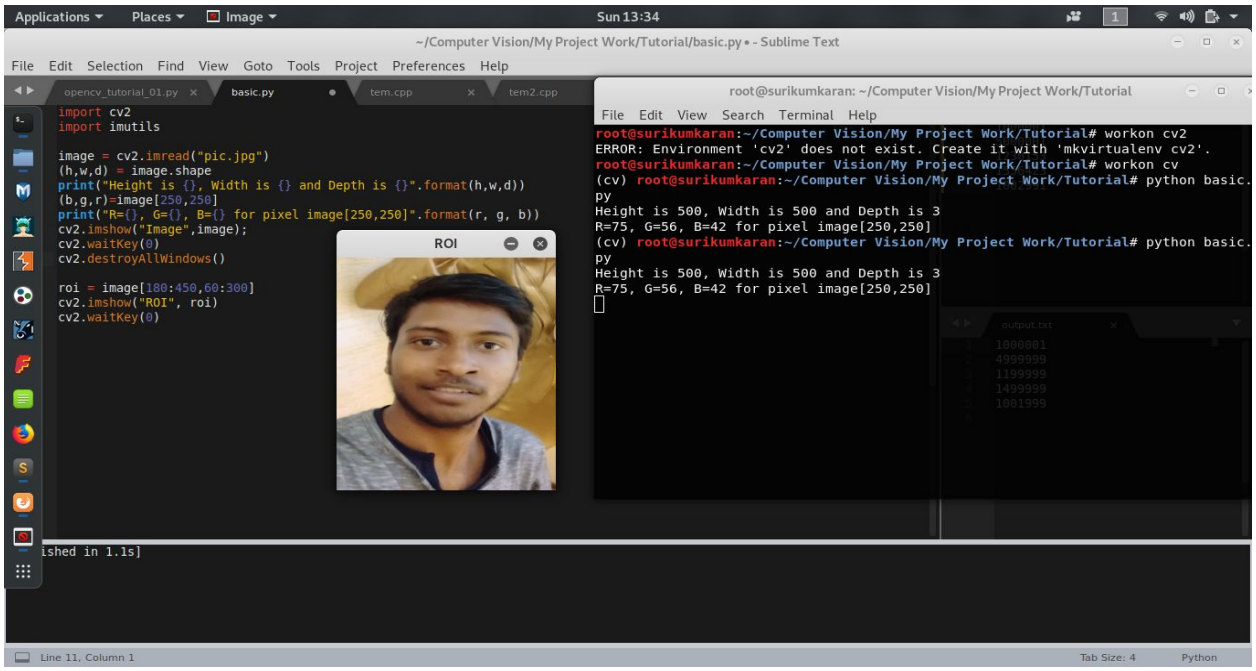
Things Which I have Learned

Basics:

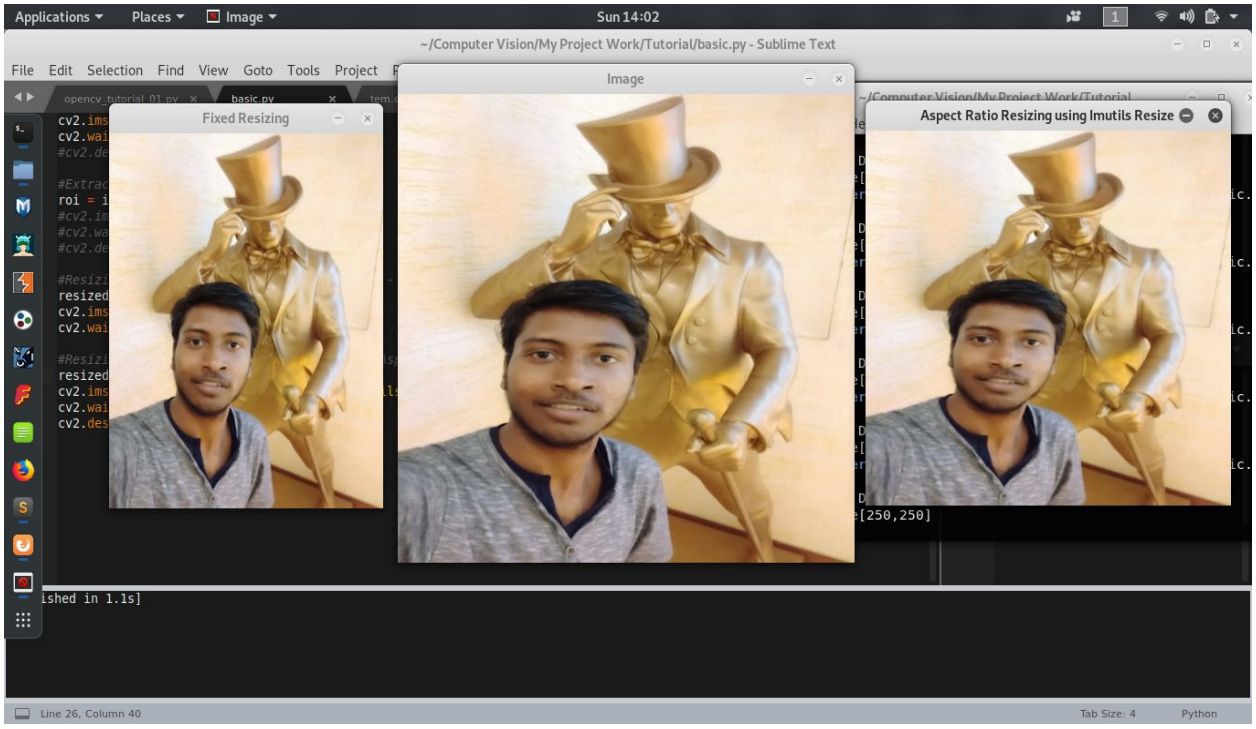
1. Loading Image and accessing shape of image and pixels.



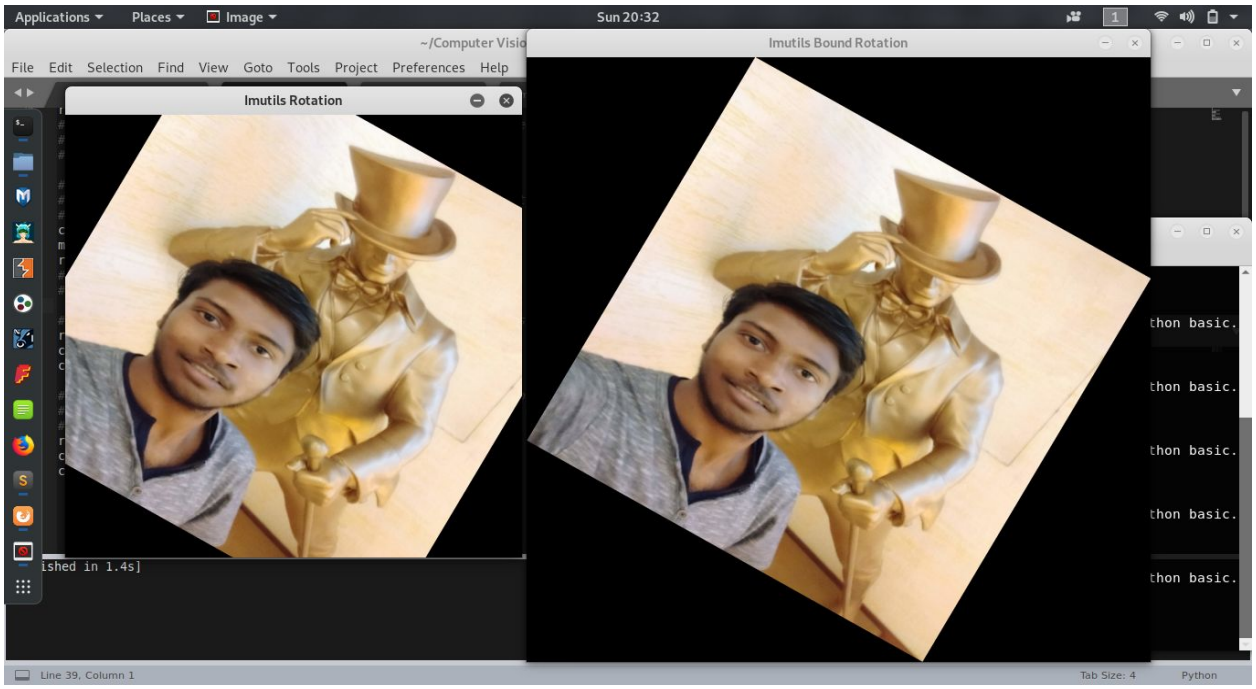
2. Extracting Region of Interest



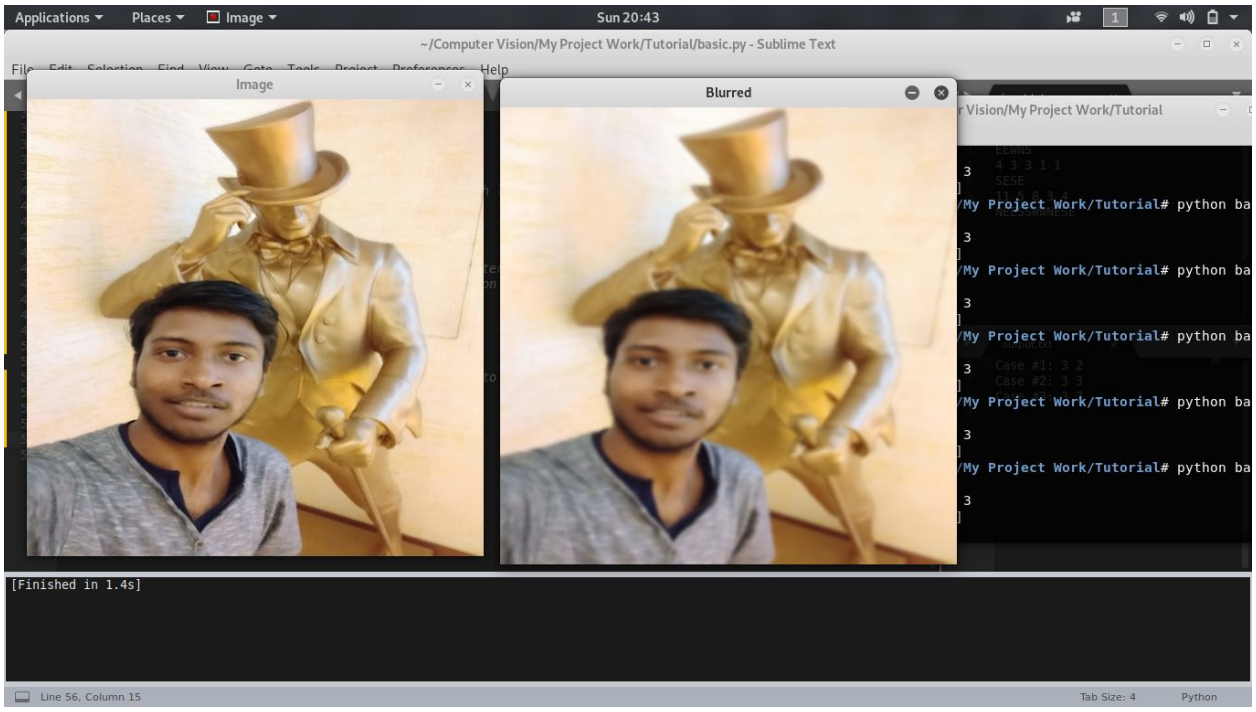
3. Resizing Image - Fixed Resizing, Aspect Ratio Resizing



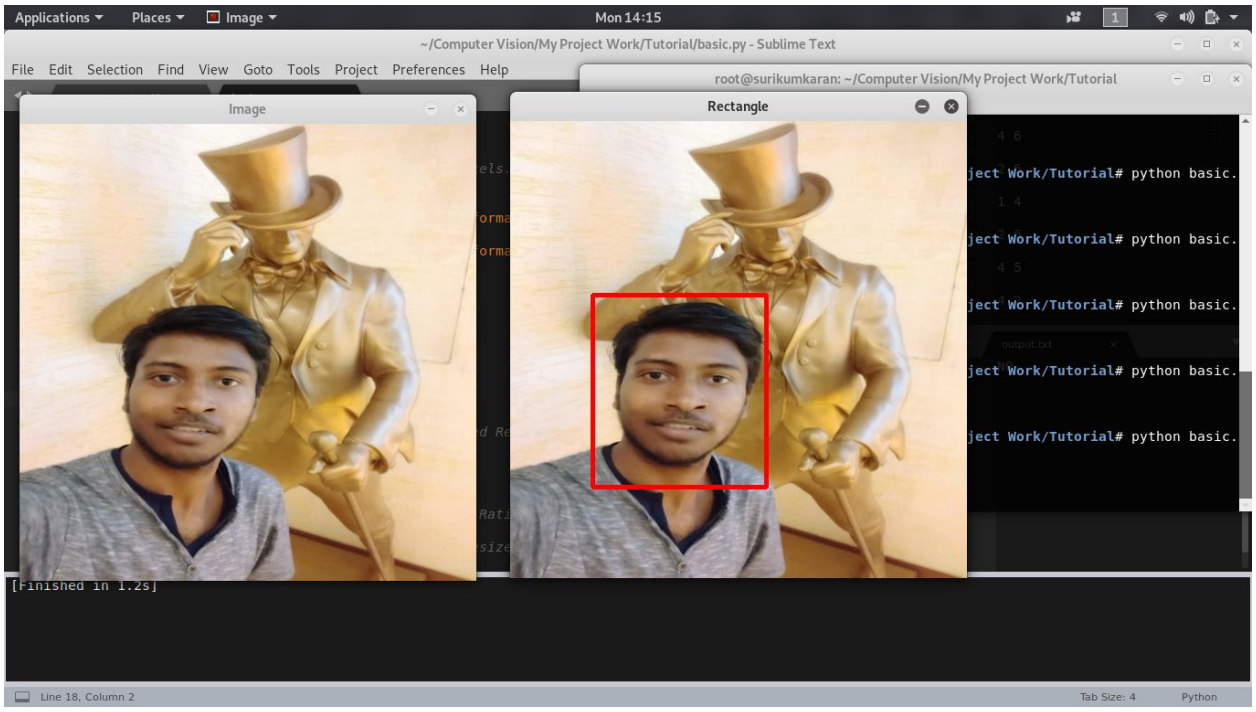
4. Rotating Images - Clipped Rotation, Bound Rotation



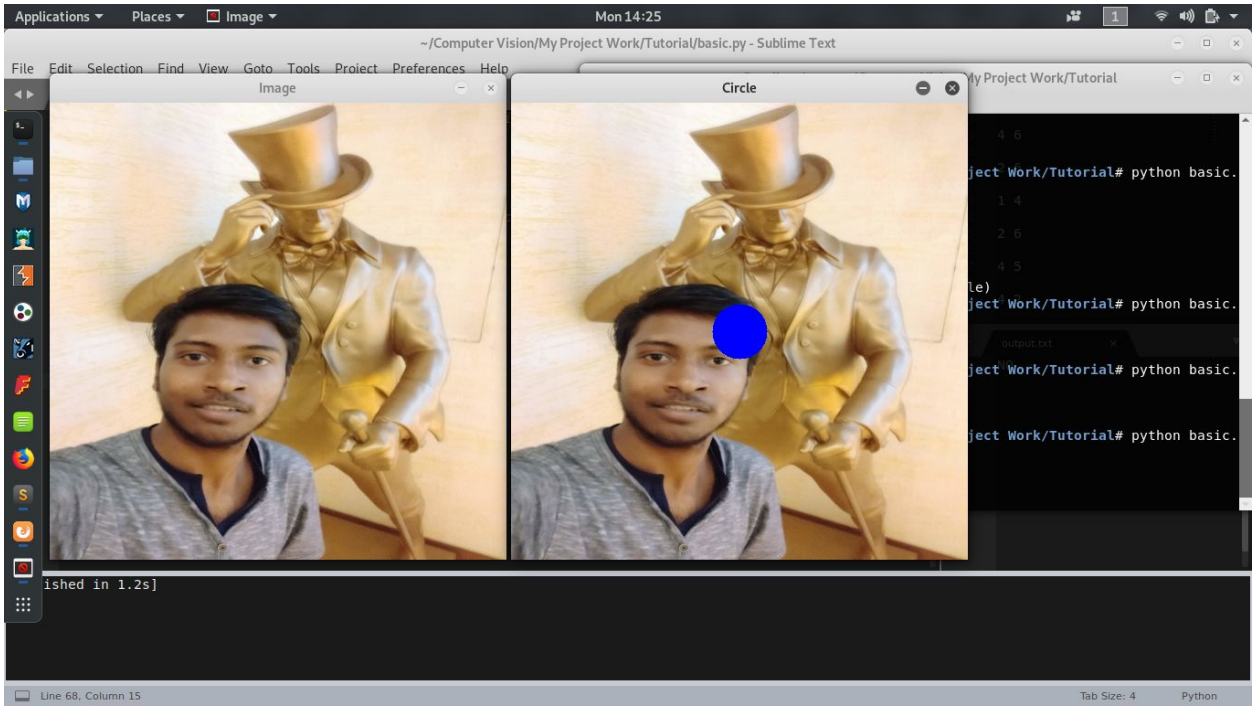
5. Smoothing Image using Gaussian Blur



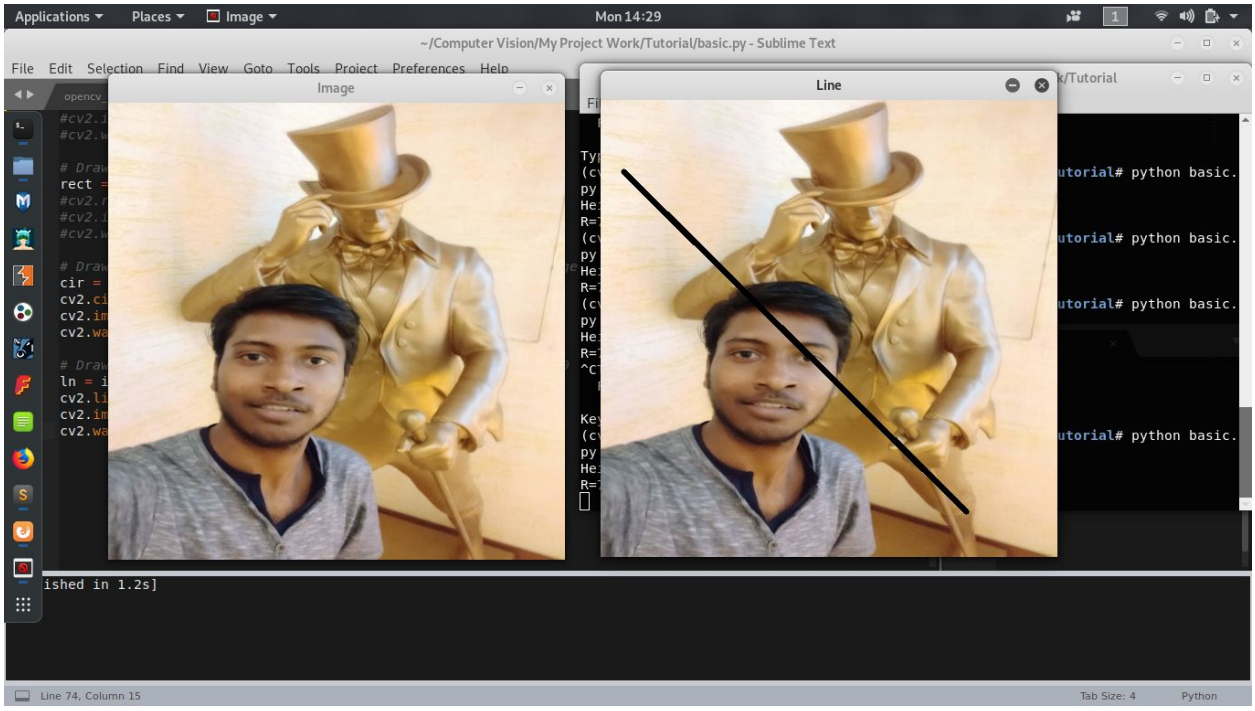
6. Drawing Rectangle



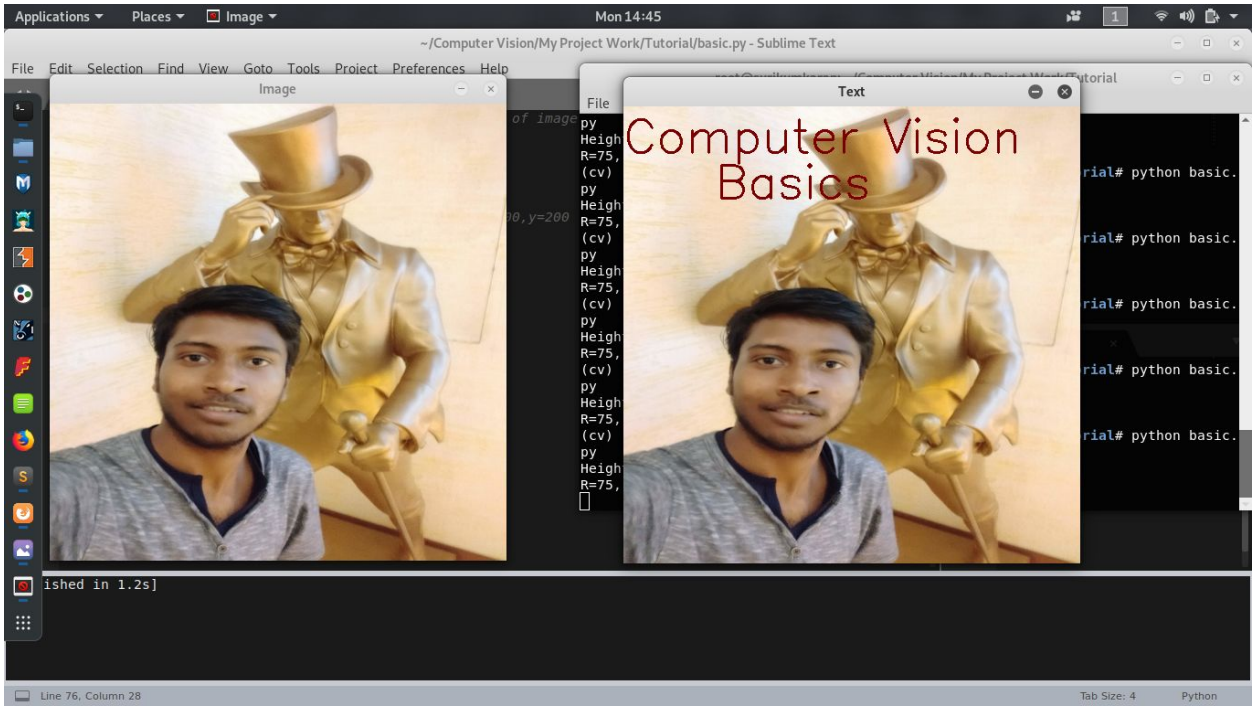
7. Drawing Circle



8. Drawing Line

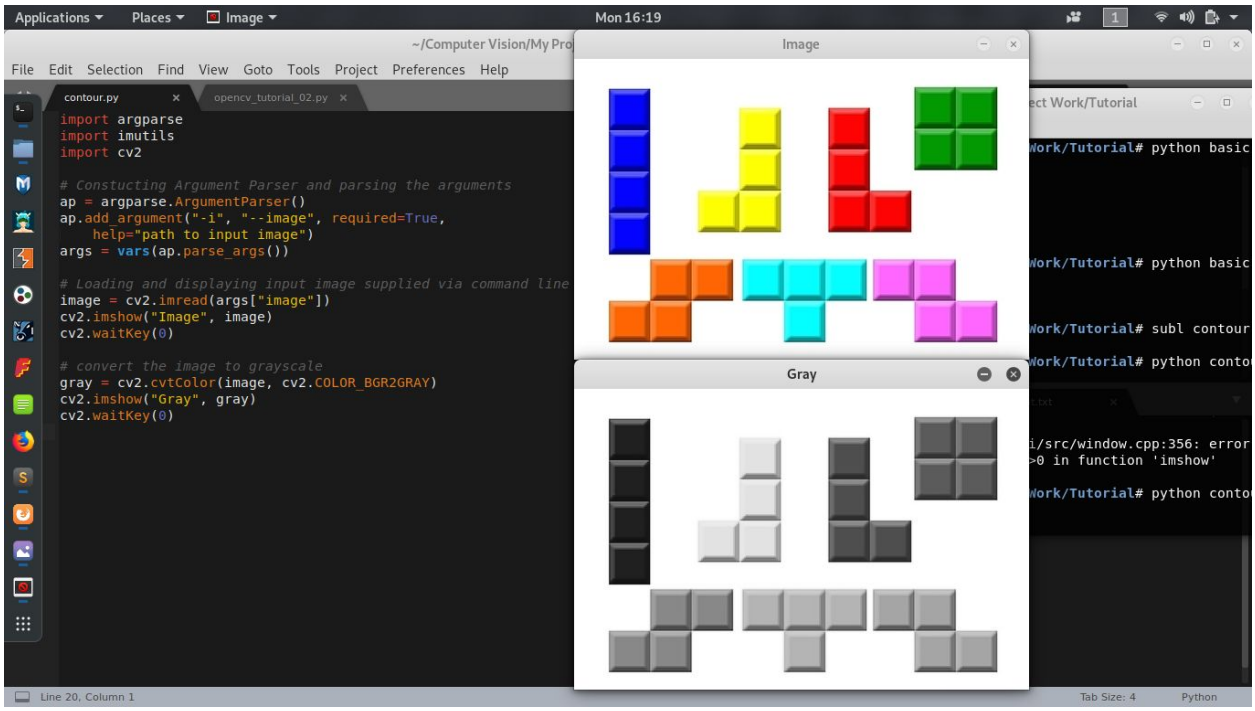


9. Drawing Text

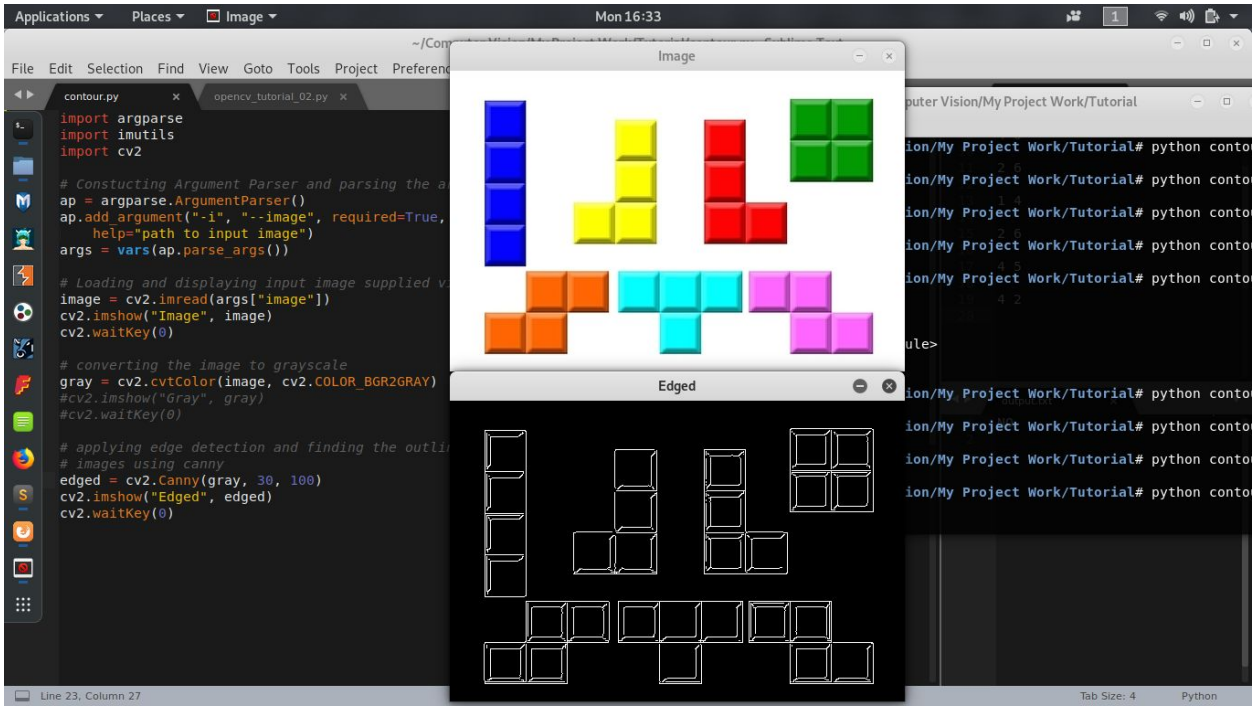


Counting Contours:

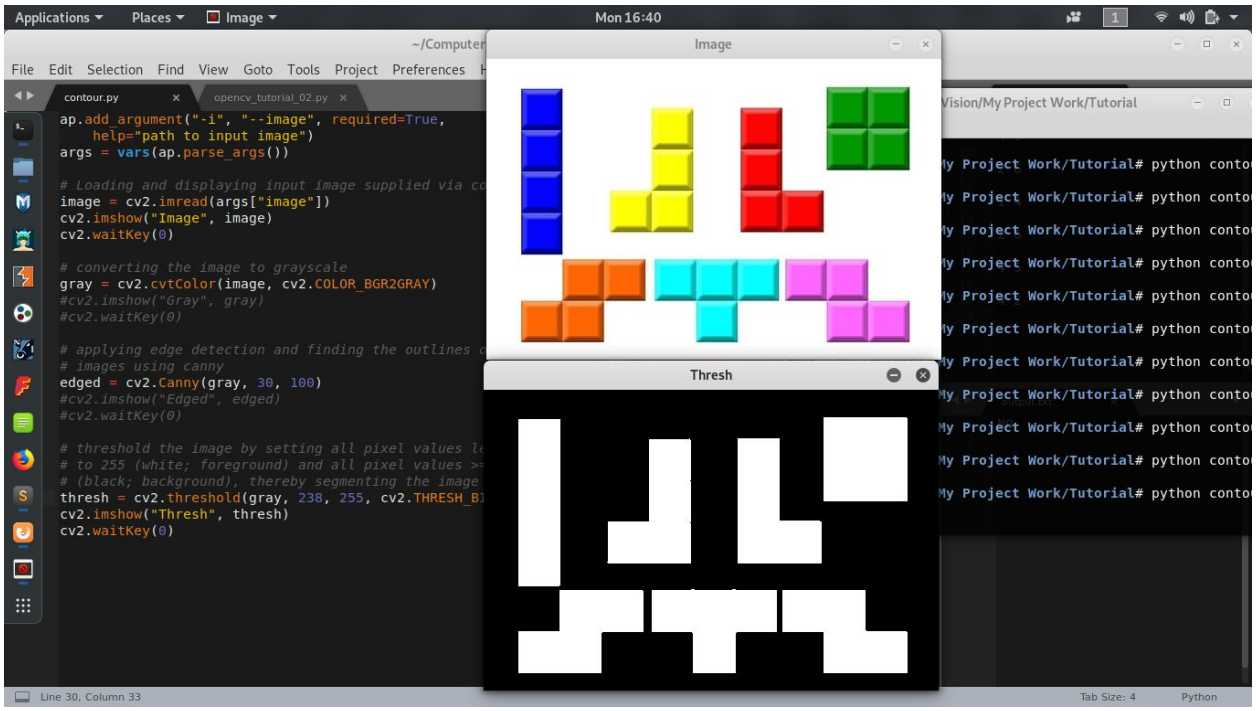
1. Converting Image to Grayscale



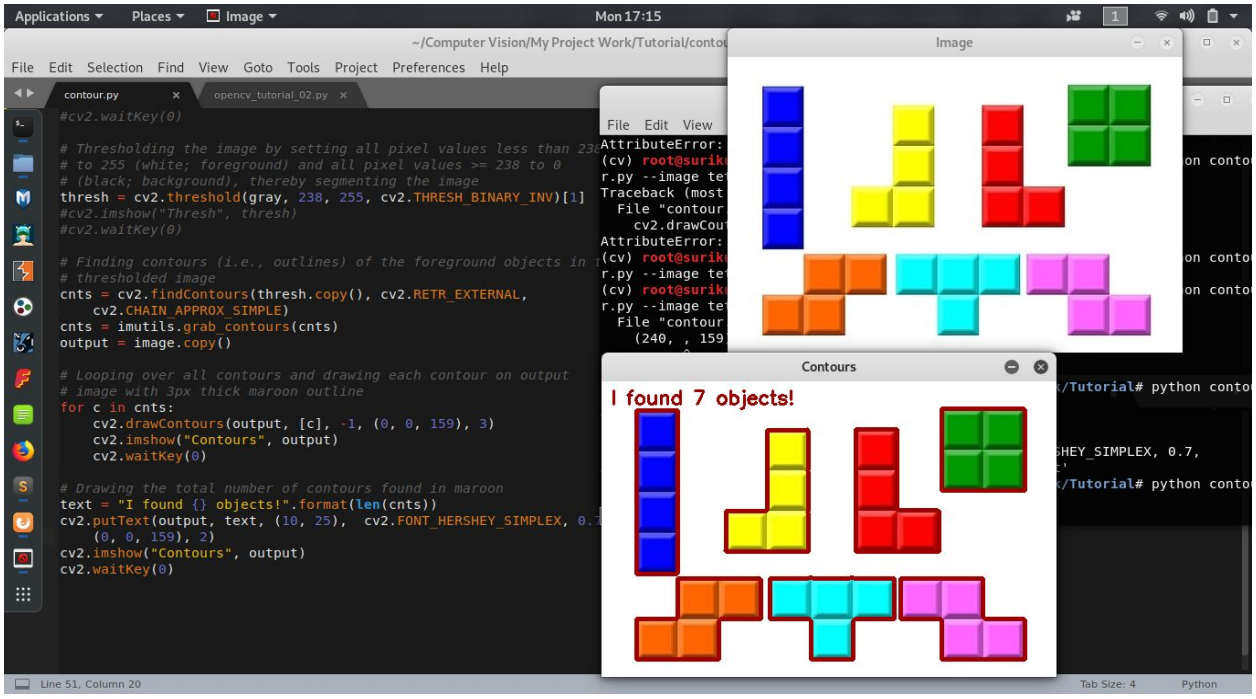
2. Edge Detection by optimizing parameters of cv2.Canny()



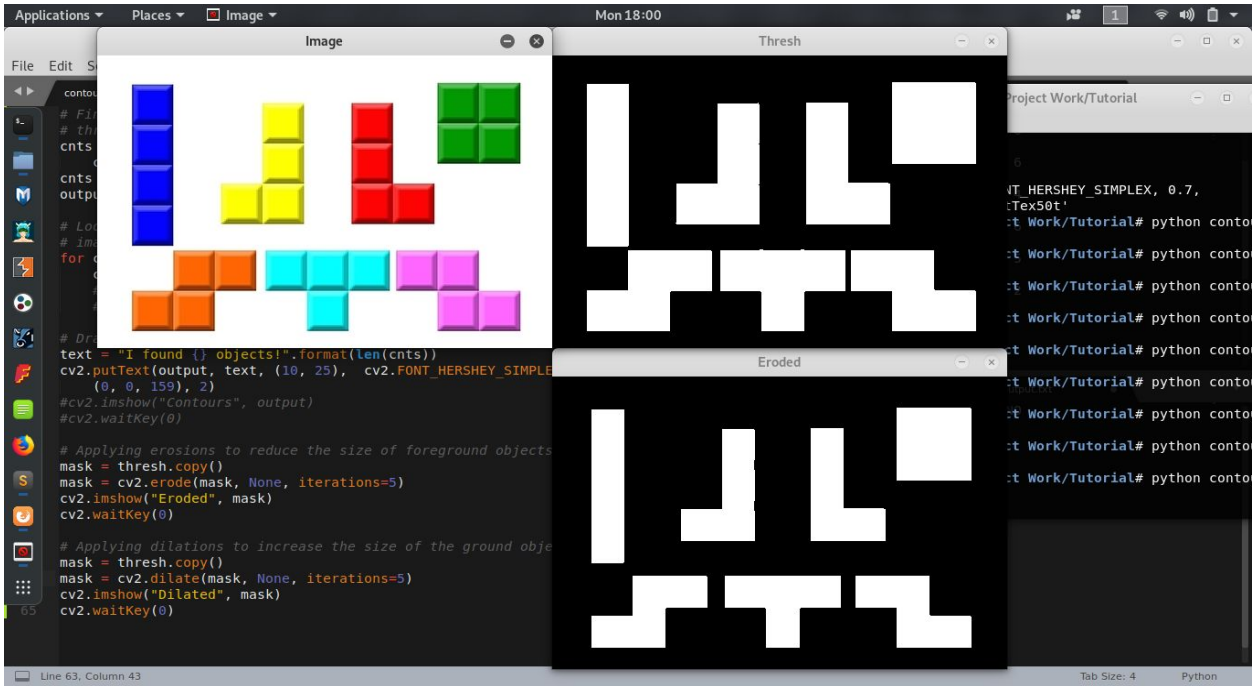
3. Thresholding Image by optimizing parameters of cv2.threshold()



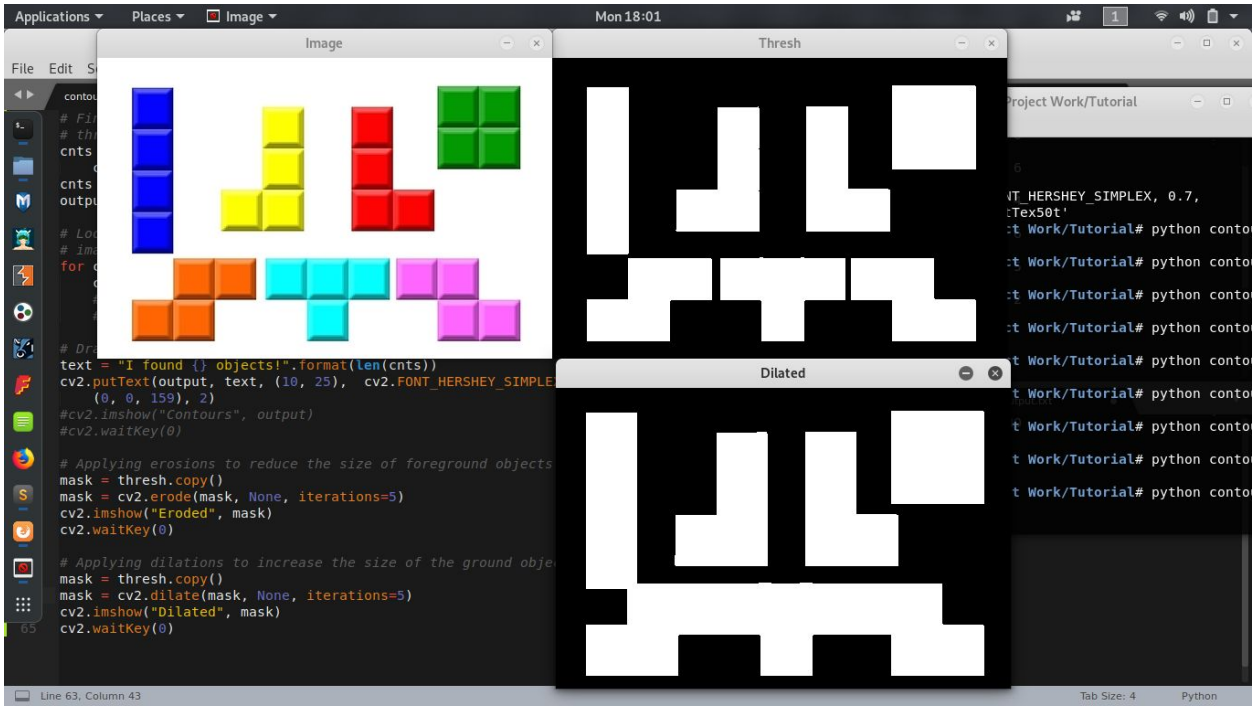
4. Finding and Drawing Contours from Threshold Image using findContours() and drawContours()



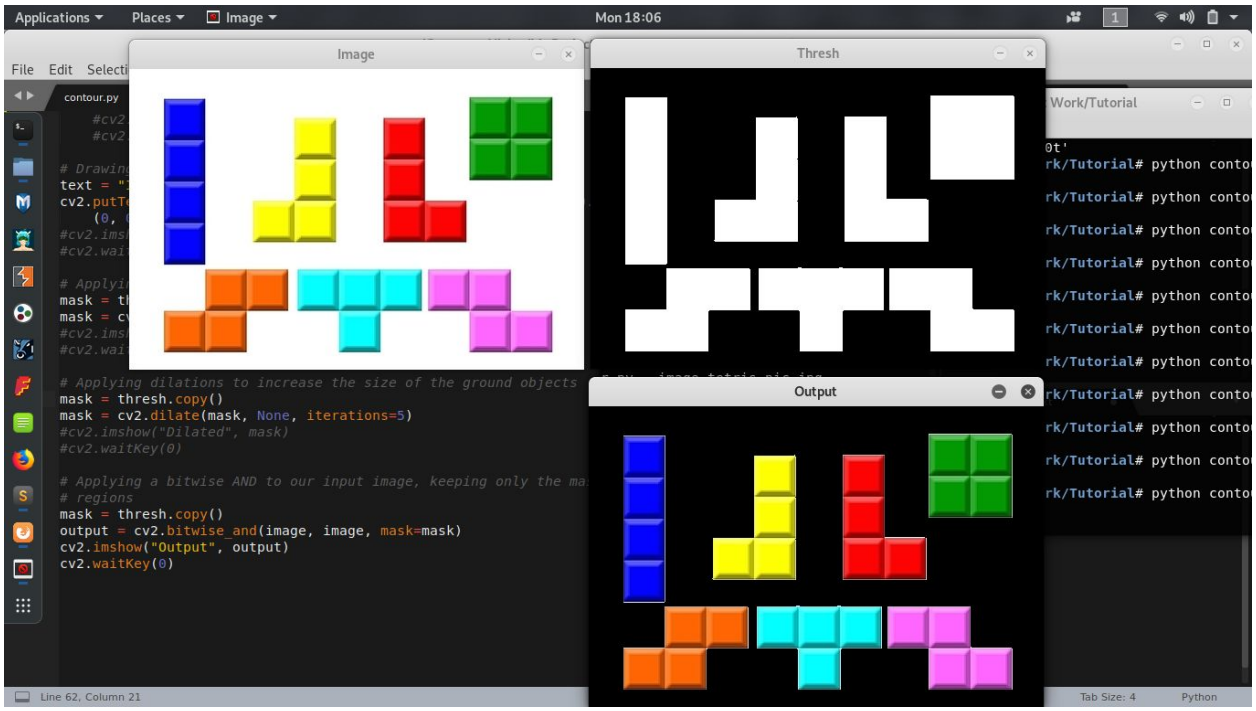
5. Applying Erosions



6. Applying Dilations

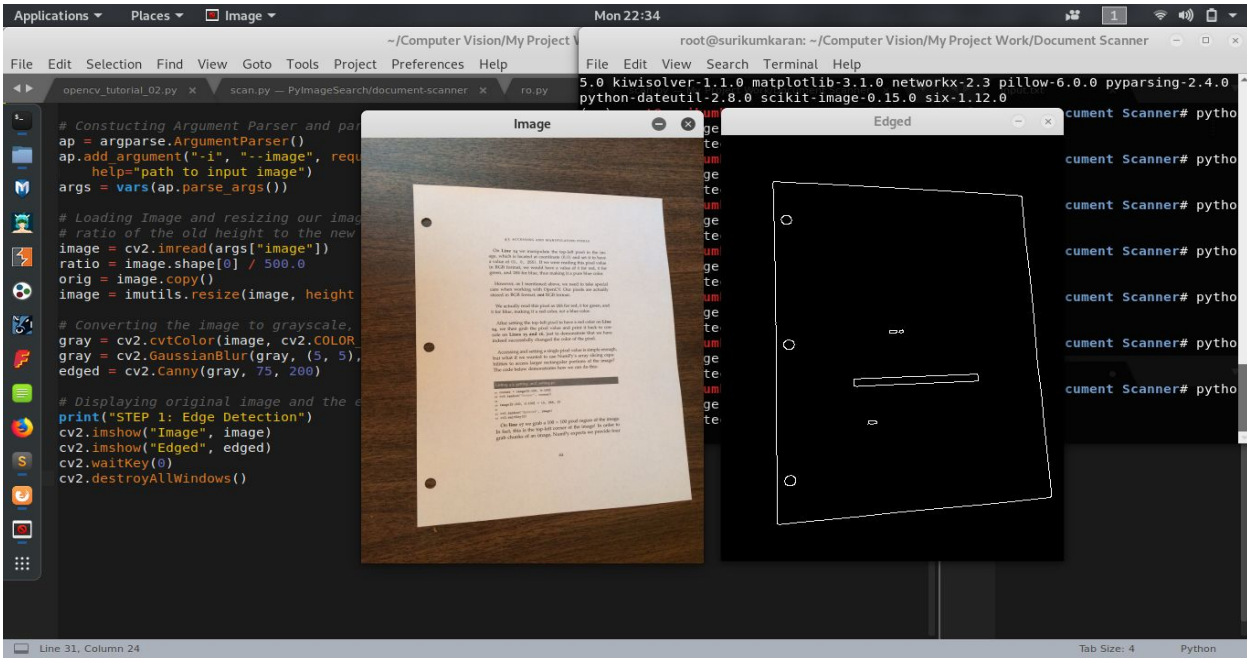


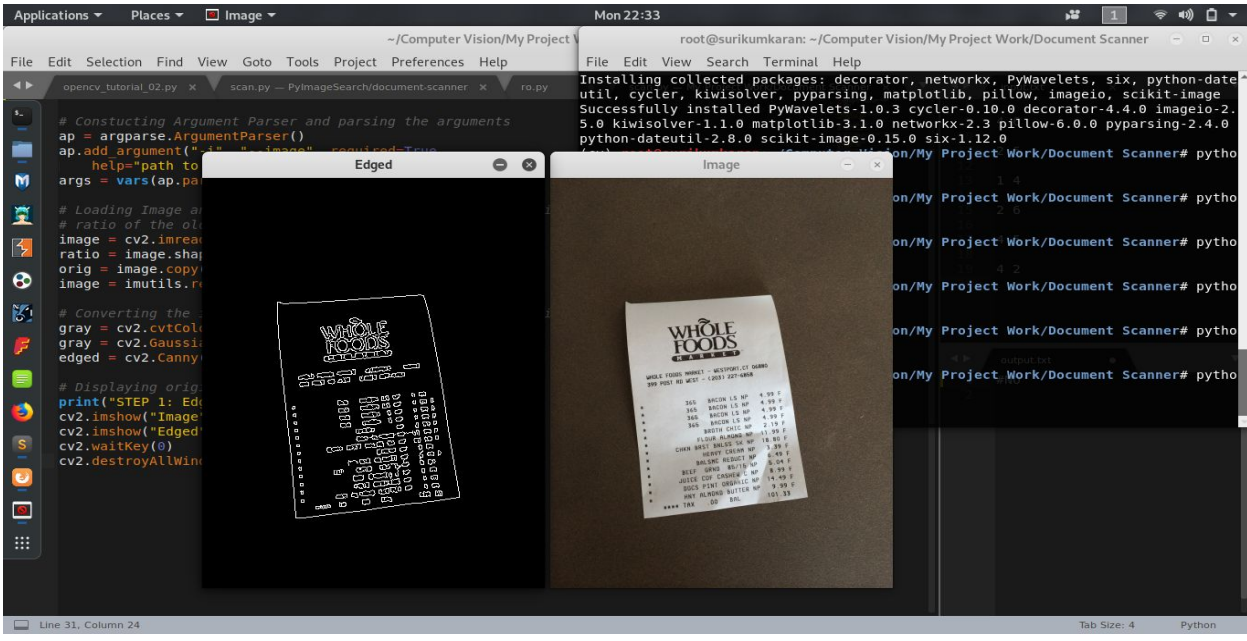
7. Applying Bitwise AND to input image using only the masked regions



Document Scanner :

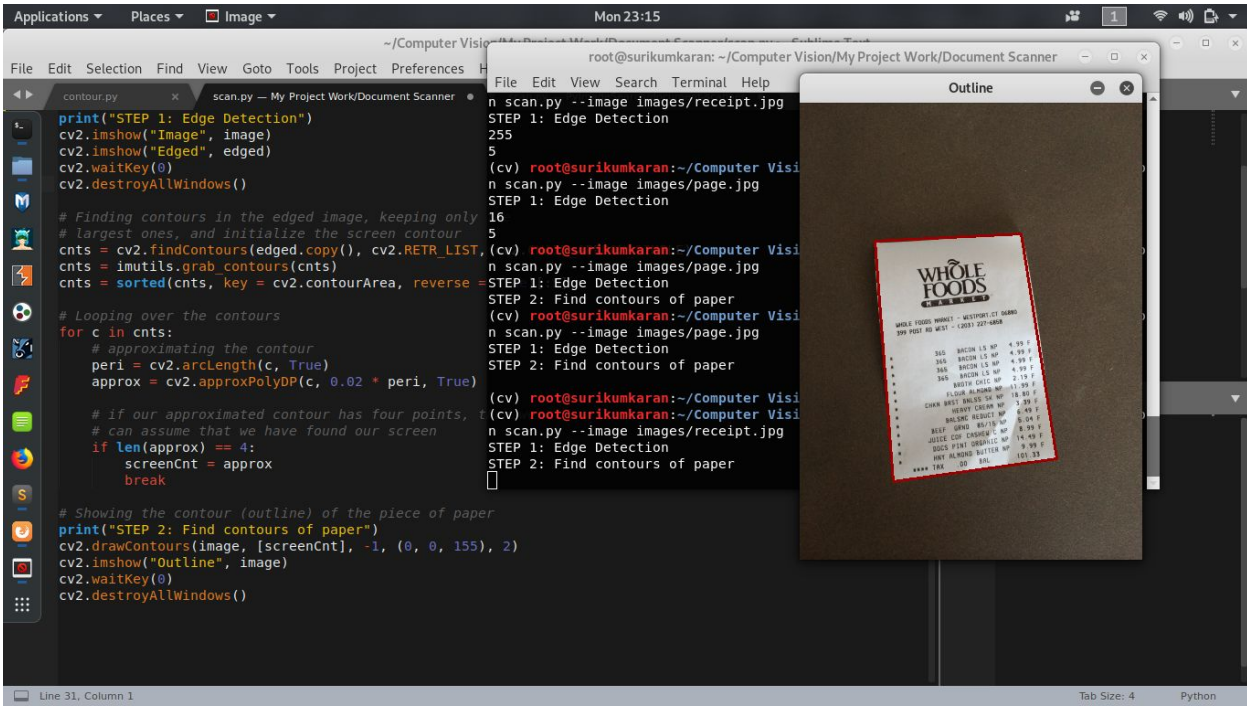
1. Edge Detection
    - Resizing our image to work faster, storing ratio of the old height to the new height.
    - Converting Image to Grayscale and then applying Gaussian blur then finding edges.

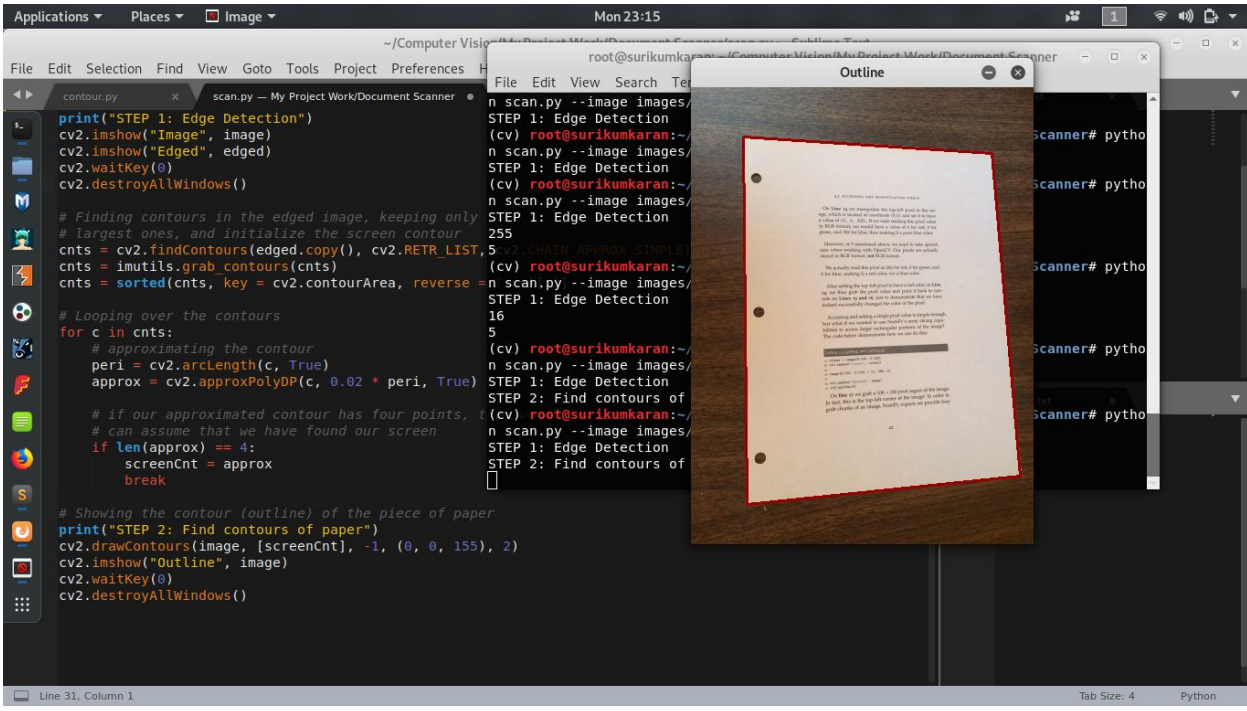




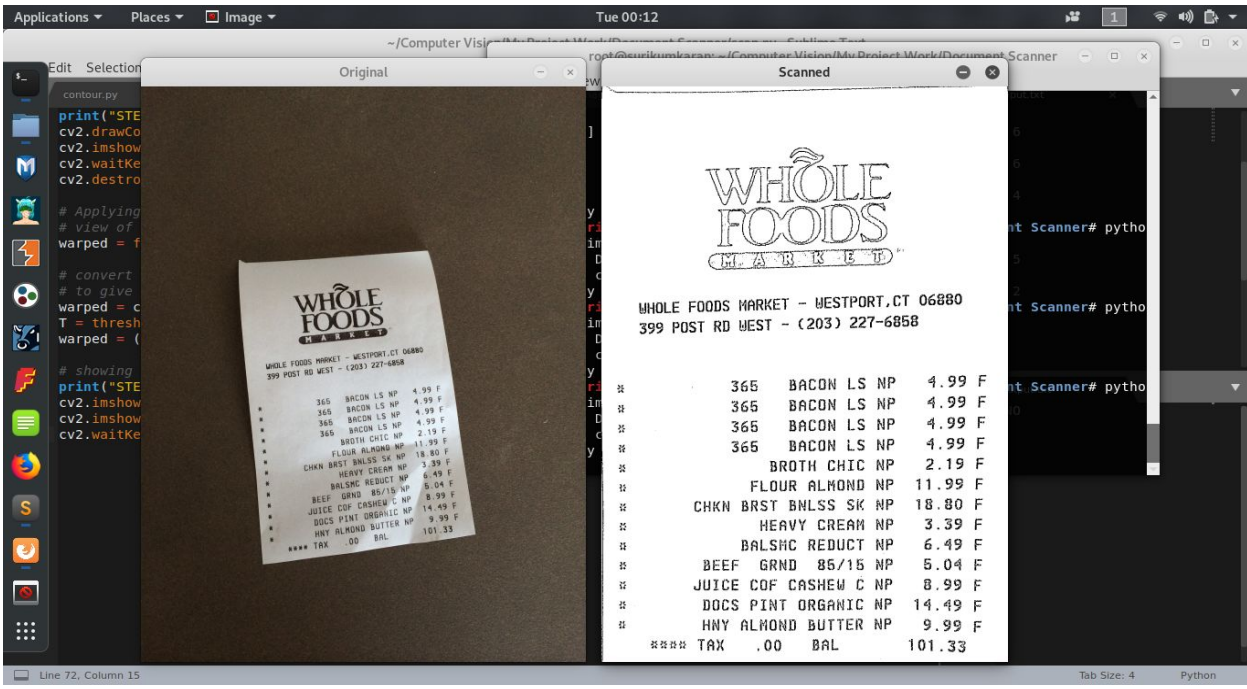
2. Finding Contours of Paper

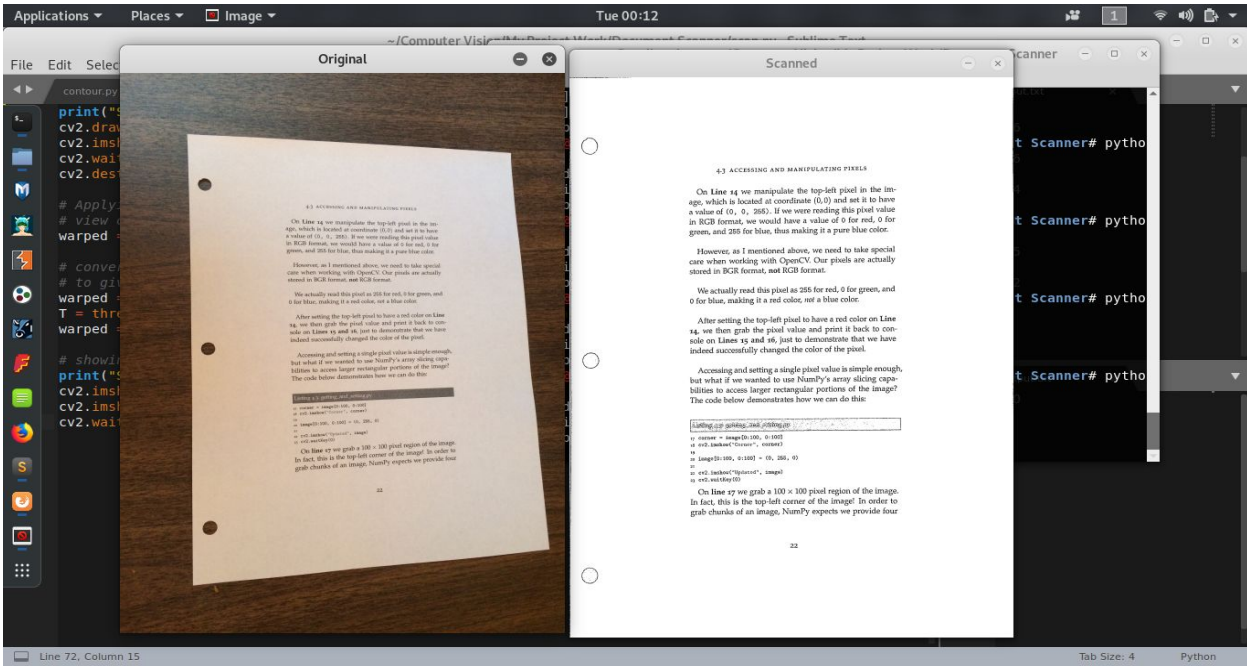
- Finding contours of edged image and then keeping only contours having maximum area by sorting in descending order by size of contours.
- Looping over contours having maximum areas and approximating them and if it has 4 points then it's our required contour.





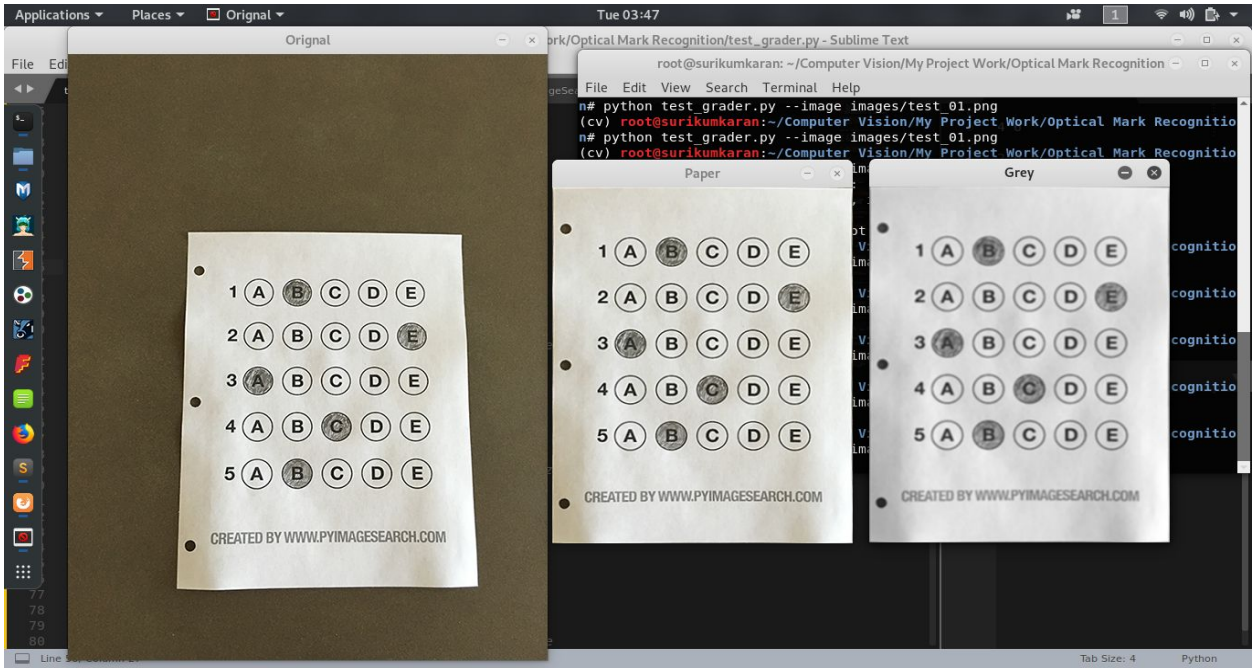
3. Applying perspective transform and Threshold
    - Applying the four point transform to obtain a top-down view of the original image using four_point_transform() of imutils library.
    - Converting it to Grayscale and then applying threshold_local from skimage.filter



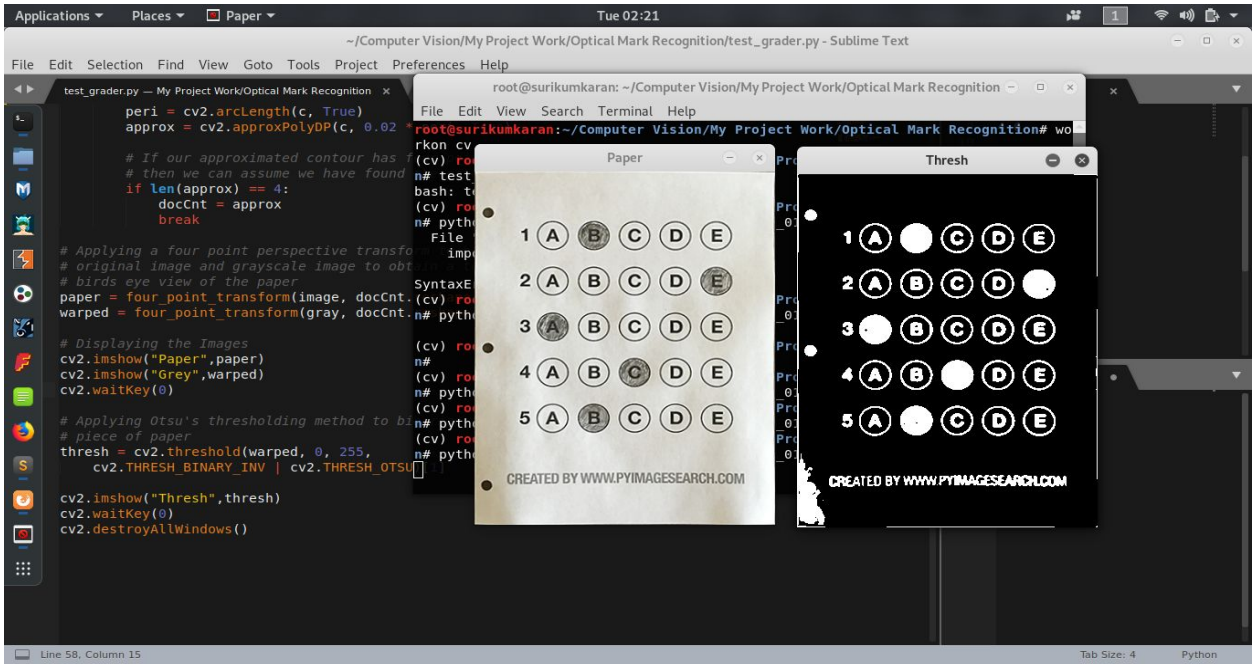


Bubble Sheet Scanner and Grader :

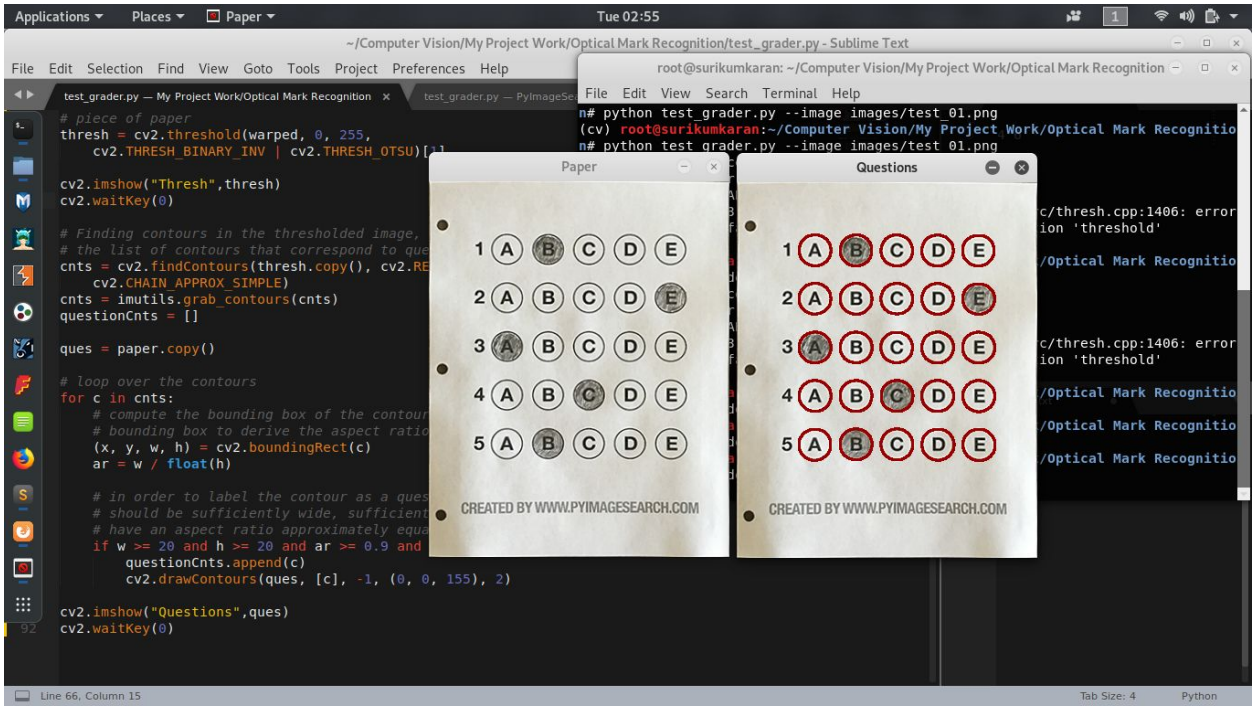
1. Scanning Document and applying Perspective Transform on Image and Grayscale Image



2. Thresholding Grey Image using Otsu's thresholding method to binarize the warped piece of paper.



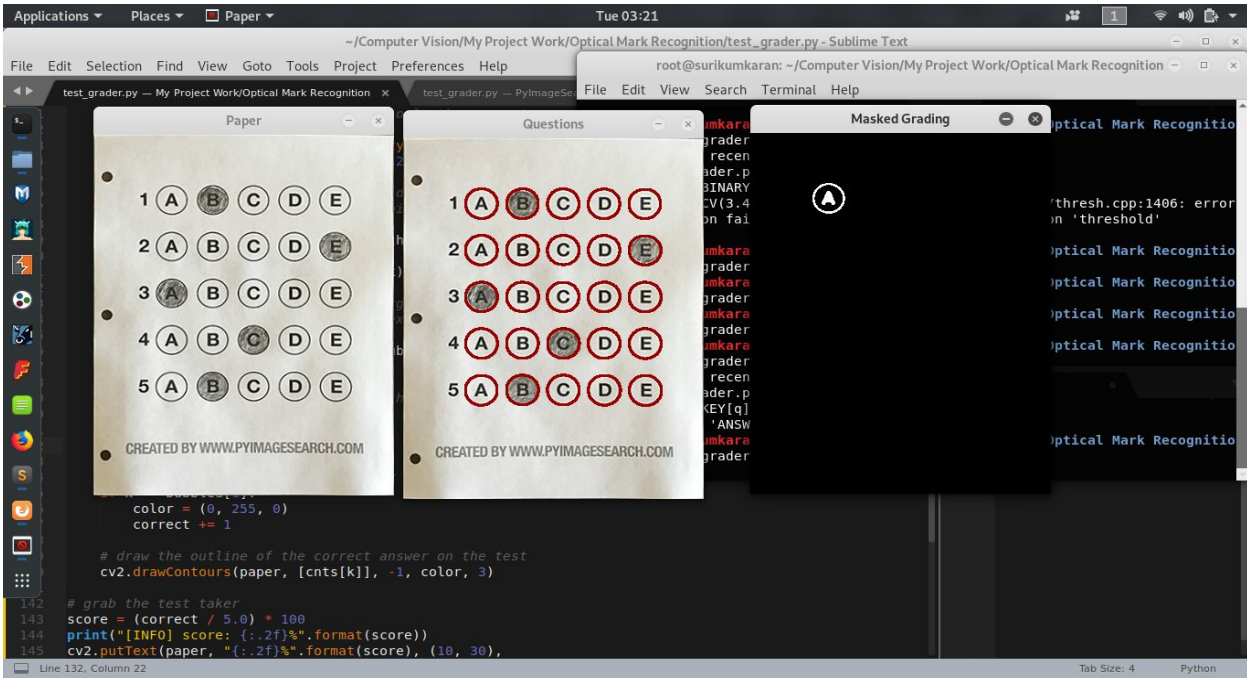
3. Finding Contours of bubble in binary warped paper. In order for a contour area to be considered a bubble, the region should :
    - Be sufficiently wide and tall (in this case, at least 20 pixels in both dimensions).
    - Have an aspect ratio that is *approximately* equal to 1.

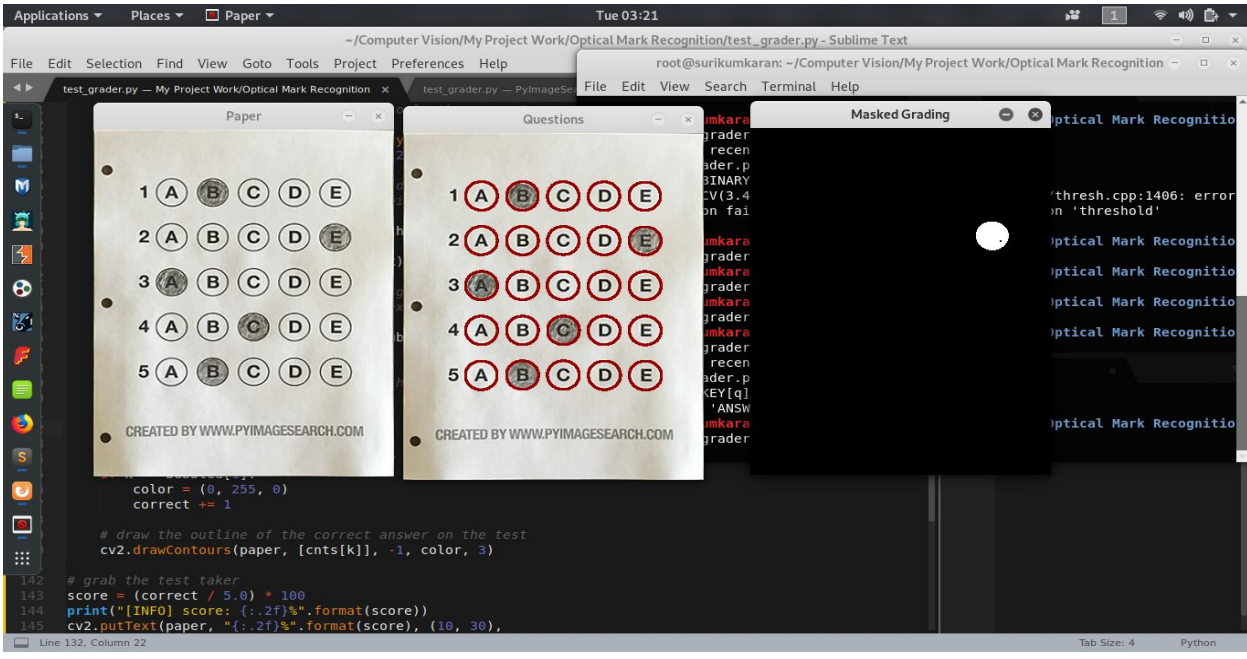


4. Arranging Contours of Bubbles :
    - First, we must sort our list of bubble contours from top-to-bottom. This will ensure that rows of questions that are *closer to the top* of the exam will appear *first* in the sorted list.
    - Then we will sort from left to right in batch of 5, as we have 5 options for each question.

5. Grading Bubbles :

- We then construct a mask for the current bubble and then count the number of non-zero pixels in the masked region
- The more non-zero pixels we count, then the more foreground pixels there are, and therefore the bubble with the maximum non-zero count is the index of the bubble that the the test taker has bubbled in.





6. Final Marking and Drawing Answer Contours