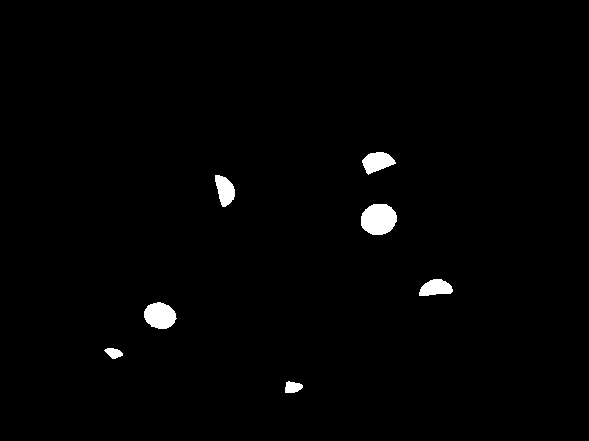
MolecuLight Coding Challenge

The first four lines are the imports used

The function **readImage** simply reads the image from the directory and returns the image as well as its hsv.

The function **highlightCircles** finds and highlights the circles.  
It has 2 constant arrays, lower and upper, which are the lower and upper RBG values of yellow range, respectively.

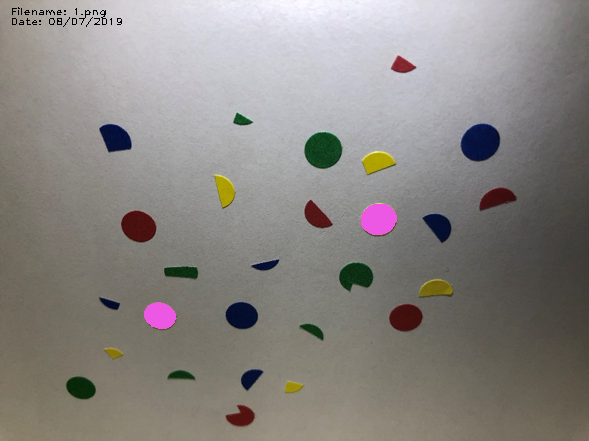
Next, we find the objects in the image that fall in those color category and mask them, as shown below:



Uncomment the three commented out lines to see the masked image

Then inside the for loop, we determine the shapes of the items we found, we use contours and filter using the number of vertices. I assumed that anything with more than 6 vertices will be a circle, highlight those shapes and return the highlighted image

The function **writeImage** writes the highlighted image in the directory with its filename and date:



The function **verify** verifies if the correct number of items were found.

It asks the user for the number of items to expect

Then it masks the output and finds the items that fall in the highlight color range.

It returns true if the correct number of items were found, false otherwise.

In order to make it realtime, we would need to keep analyzing a set of images at a very fast speed.

The biggest trade offs would be, we would need more powerful hardware, thus more energy would be required. We would also need larger storage to store the images

And given that we are looking for speed we would lose accuracy of our identification, and some items might be misidentified or not identified at all in some frames.

References:

1. <https://www.pyimagesearch.com/2018/07/19/opencv-tutorial-a-guide-to-learn-opencv/>
2. <https://codeloop.org/opencv-python-color-detection-example/>
3. <https://www.pyimagesearch.com/2014/07/21/detecting-circles-images-using-opencv-hough-circles/>
4. <https://docs.opencv.org/3.3.1/d4/d73/tutorial_py_contours_begin.html>
5. <https://datacarpentry.org/image-processing/09-contours/>

Note:  
Since I do not have access to a windows machine, I was not able to create a windows exe file.

However, I did create a bash executable file: run.sh

In the current code, the user is asked for the name of the image, as well as the number of items for the verify function.  
But if you uncomment lines 86 and 93 in highlight.py as well as line 4 in run.sh, and comment out lines 85, 92 in highlight.py as well as line 3 in run.sh, and execute run.sh, it will run the same code without asking for user inputs

I put the masked images as well as my output that I got from the code as reference to the working function

Please email me if you have further questions, and I can always come in and give a quick demo of my code