Required Python Libraries:

i. NumPy
ii. OpenCV
iii. dlib
iv. imutils

Object Detection and Object Tracking

When we apply object detection we are determining where in an image/frame an object is.
An object tracker, on the other hand, will accept the input (x, y)-coordinates of where an object is in an image and will:

1. Assign a unique ID to that particular object
2. Track the object as it moves around a video stream, predicting the new object location in the next frame based on various attributes of the frame (gradient, optical flow, etc.)

Combining both object detection and object tracking

Highly accurate object trackers will combine the concept of object detection and object tracking into a single algorithm, typically divided into two phases:

Phase 1 — Detecting: During the detection phase we are running our computationally more expensive object tracker to (1) detect if new objects have entered our view, and (2) see if we can find objects that were "lost" during the tracking phase. For each detected object we create or update an object tracker with the new bounding box coordinates. Since our object detector is more computationally expensive we only run this phase once every N frames.

Phase 2 — Tracking: When we are not in the "detecting" phase we are in the "tracking" phase. For each of our detected objects, we create an object tracker to track the object as it moves around the frame. Our object tracker should be faster and more efficient than the object detector. We'll continue tracking until we've reached the N-th frame and then re-run our object detector. The entire process then repeats. We use **Kalman filtering** for object tracking.

The benefit of this hybrid approach is that we can apply highly accurate object detection methods without as much of the computational burden. We will be implementing such a tracking system to build our people counter.`

Zeroing is one of the most important two directories, we have:
i. Bottle: This module contains **centroid tracking algorithm.**
2. mobilenet_ssd: Contains the caffe deep learning model files. We will be using **Mobilenet SSD** as single shot detector for object detection.

To implement our people counter we'll be using both OpenCV and dlib. We'll use OpenCV for standard computer vision/image processing functions, along with the deep learning object detector for people counting.

Combining object tracking algorithms

To implement our people counter we'll be using both OpenCV and dlib. We'll use OpenCV for standard computer vision/image processing functions, along with the deep learning object detector for people counting.

We'll then use dlib for its implementation of correlation filters. We could use OpenCV here as well; however, the dlib object tracking implementation was a bit easier to work with for this project.

Along with dlib's object tracking implementation, we'll also be using centroid tracking algorithm.

Steps:
Step 1: We accept a set of bounding boxes and compute their corresponding centroids (i.e., the center of the bounding boxes). The bounding boxes are provided by the correlation filters.
Step 2: We compute the Euclidean distance between any *new* centroids and existing centroids :
The centroid tracking algorithm makes the assumption that pairs of centroids with minimum Euclidean distance between them *must be the same object ID*. Computation is thus done to compute the distance between those centroids.
Step 3: Once we have Euclidean distance, we attempt to associate object IDs. If we find other centroid that cannot be associated, then
Step 4: We register new objects by assigning it with new object id and storing the centroid of the bounding box coordinates for new object.
Step 5: Deregister the object if the object has left the field of view and has been lost.

Running command basic:
python people_counter.py -p /home/manish/Downloads/people-counting-opencv/mobilenet_ssd/MobileNetSSD_deploy.prototxt -m /home/manish/Downloads/people-counting-opencv/mobilenet_ssd/MobileNetSSD_deploy.caffemodel -i /home/manish/Downloads/people-counting-opencv/videos/labim_feed.mp4 -o /home/manish/Downloads/people-counting-opencv/output.mp4 -c 0.8