For the assignment to run the code enter required input whenever prompted.
All the code is well commented, look at the scripts/notebook code for more information.

Q1:
This question requires us to compute the quality score of two images using two different methods and tell which one of them is better.
I have asked the user to input a path to the two images, and the code finally prints which image has better saliency quality. For the test I have included two images found from the internet.
After finding the final scores, I tested the scores for many online images. I found that the score provided using the second image was more reliable(as I could see the images, and say which one is obviously better) Therefore for final quality I have taken the weighted average of the quality score I found using both methods.
Part 1:
Simply implemented the equations given in the paper provided. I have not made a histogram at any point as I simply created an array of two images based on the threshold values. Using the arrays I counted the number of pixels in the desired range(integration part). To get the intersection point I took the coefficients of the equation and put them in predefined function.
Finally returned the quality according to the formula

Part 2:
Got the connected components using the function. The function also gave stats on the components(including area). Used these stats to find the desired quality according to the equations provided.
Finally compared the quality of the images and printed which image was better in quality

Q2
Wrote a function to get the lbp according to the question.Then wrote a function to use these lbp values of the image to get the spp. To get the SPP I divided the images in 16, 4 and 1 part. Got histogram for each of them and concatenated them to get the SPP.
Got spp for all the images in the dataset. Fit them using k means.
Used the clusters of the kmeans to get the clusters. Created folders according to the clusters formed, finally zipped them as the final output(did in notebook, not in python).(python script will produce folders only)

Q3:
Created a function which takes a path to the image and sends the hog features(using skimage.feature.hog method). Found these features for all images and stored them to a variable named finallist.used k means to this least(after flattening them)
Ask the user to input the path to the image for which feature vector is required. Get the features.
Use the kmean model trained above to get the cluster to which each feature belongs.Finally count the number of features belonging to each cluster.
Finally return this feature vector(which is our final ans)

Q4:

Created a function(findfeatures) to get the lbp value of the top 100 corner points in the image. I sorted the values so that it is easy to compare this vector with that of the other.
I stored this feature vector of all the images in the dataset.
Asked to input the path to the image which we want to read. Got the feature vector of it.
Compared this feature vector with all other vectors in the dataset(used distance to compare).
Finally printed names of the nearest k images.