## Question 1

All the codes were written in colab. They are written according to colab.

### Color AutoCorrelogram:

1. I am changing the size of all the images to 500*500*3
2. D = [1,3,5,7]
3. I am using my own cluster algorithm as is we use a separate k means for all the images, it forms separate clusters which cannot be compared.
   In this the pixel values are converted to integers 0,1,2,3 based on their values.
   if(pixel < 64 ) → converted to 0 and so on.
4. I am taking 8 neighbours in each case

```
p1 = (x+d,y+d)
p2 = (x-d,y-d)
p3 = (x+d,y)
p4 = (x-d,y)
p5 = (x,y+d)
p6 = (x,y-d)
p7 = (x+d,y-d)
p8 = (x-d,y+d)
```

5. The similarities of the colours are matched with its neighbours and this is used to make the color correlogram
6. A separate pickle file is made for all the the images and stored under the folder PickleFile_Question1
7. These all pickle files are opened and made collectively into 1 pickle file. This pickle file is in folder All_PickleFiles_Question1

**Results:**

The results are not very good. They have low precision and recall. This can be because of the poor clustering method or due to resizing of the image. Both these things were necessary to be done to reduce the time for computation. To analyse the results we are computing the difference in the correlograms of the query image and all the other images. Then we are calculating the 2 norm of the matrix and calling it as the distance. Then the 100 images with the minimum distance are taken.

```
Max Precision=  0.26
Min Precision=  0.01
Average Precision=  0.04151515151515151526
Max recall=  0.214285714285714427
Min recall=  0.015625
Average recall=  0.07015842173744764
Max F1=  0.115299334811529993
Min F1=  0.012195121951219513
Average F1=  0.038733303450605960
Average percentage good=  11.667341742813438
Average percentage ok=  5.662351116896573
Average percentage junk=  2.74286987522281620
Max good=  42.857142857142854
Min good=  0.0
Max ok=  25.0
Min ok=  0.0
Max junk=  14.285714285714285
Min junk=  0.0
```

This file is run in Testing_Question1.ipynb using my other ID because I was facing some issues with GPU on this ID.
The main file making the pickle files was Question1.ipynb.

Precision - Relevant images retrieved / total images retrieved
Recall - Relevant images retrieved/total number of relevant images
F1 - HM of precision and recall

Total images retrieved - 100
Total number of relevant images - Depends upon the number of images in good/junk/ok folders.

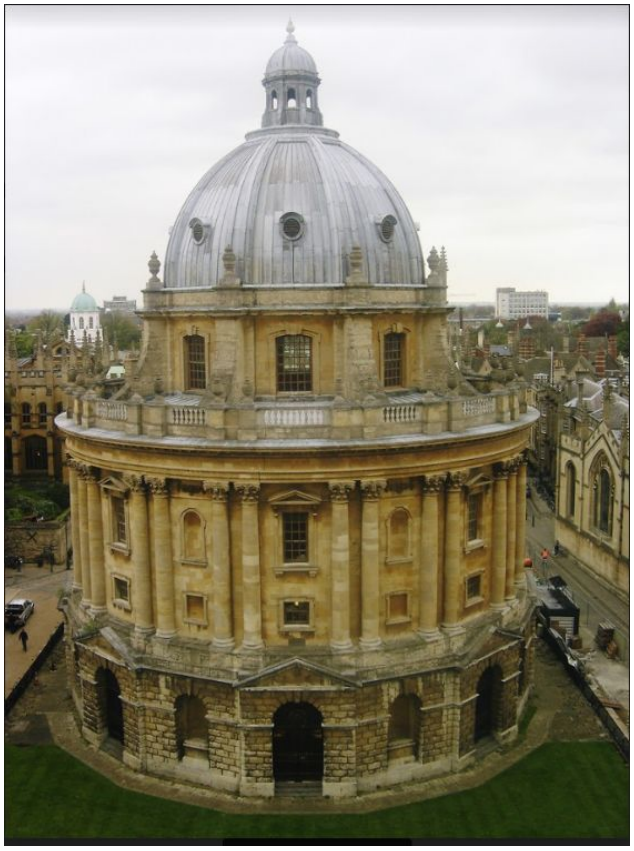
**Analysis.**

Oxford_000317

This image has the minimum precision of 0.01
Minimum recall of 0.015625
Minimum F1 of 0.0121

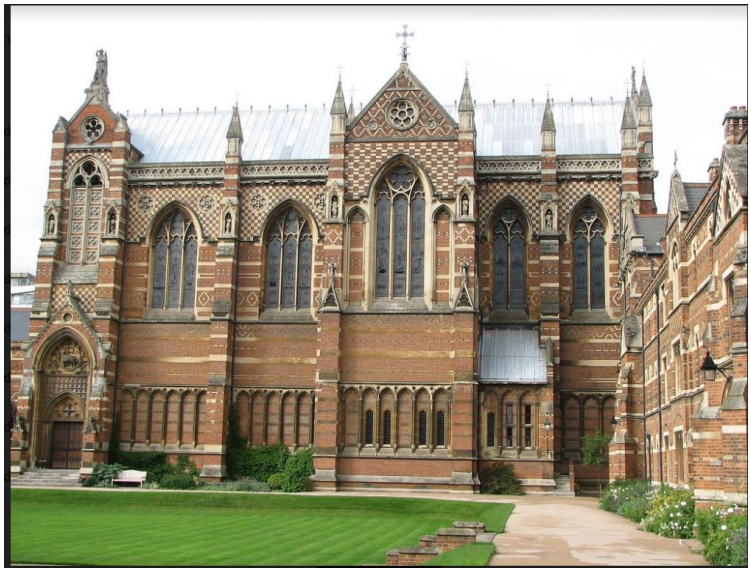
This may be because it has various patches of some other color in form of windows with make it difficult to form the correlogram. It has only 2.7 percent match with the images in the good section. And 0 percent match with the images in the ok and junk section. This can also be a main reason for less precision, recall and F1 scores. The partition between the various colours is also very light. The part of the image in the 500*500 might not be the same as the rest of the image.

Radcliffe_camera_000523



This image has the highest precision score of 0.26 and also the highest F1 score of 0.115. It also has a high recall value. This might be due to the clear distinction of the colours. And also the clarity of the image. It also has around 15% matching of the good images. And 5% matching of the OK images. This might also be the reason for high precision. The part of the image in the 500*500 is almost the same as the rest of the image.

Keble_000227



This image has the maximum matching with the good images and it also has the maximum recall value. This is due to the clear distinction of the colours and the dominance of one color. It also has high clarity of the image. The part of the image in the 500*500 is almost the same as the rest of the image.

Average time for the image is 3.5 seconds

References:
https://github.com/enthalpy-yan/Color-Texture-Shape/blob/master/lib/Descriptors.py
https://ieeexplore.ieee.org/document/4279141

### Scale Invariant Blob Detection - LoG

1. The main file run to form the pickle files is Question2.ipynb
2. ```
   level = 16
   threshold = 0.02
   initial_sigma = 1.3
   sigma_factor = 1.24
   ```
   These are the initial parameters.
3. The image is converted in gray scale and also normalized.
4. I am saving the blob location, the pixel value of centre of blob in real image, the pixel value of the centre of the blob in the gray image.
5. Then I create a sigma space according to the various sigmas and levels. The sigma space contains the convolved images. The convolution is done using the inbuilt functions of cv2. The kernel used in the convolution is based on the LoG function created. Which is based on our selection of the sigma.
6. This we consider the location of the blob based on the values of its neighbours. If it has values similar to its neighbours then it is considered in the same blob.
7. The we find the radius and the centres of the blob.
8. I am making different pickel files for each image and saving them in folders named PickleFiles_Question2_BlobLocation, PickleFiles_Question2_GrayPixel, PickleFiles_Question2_ImagePixel
9. The pickle files of all the images is combined in a single file name All_PickleFile_Question2

Average time for the image is 2 seconds
   ```

References:

https://projectsflix.com/opencv/laplacian-blob-detector-using-python/
https://github.com/eshaan90/sift-feature-detector/blob/5d909e7b3d89cdad9c7e34eb29b5ce5b5a3ad5b6/src/blob_detector_project.py#L27