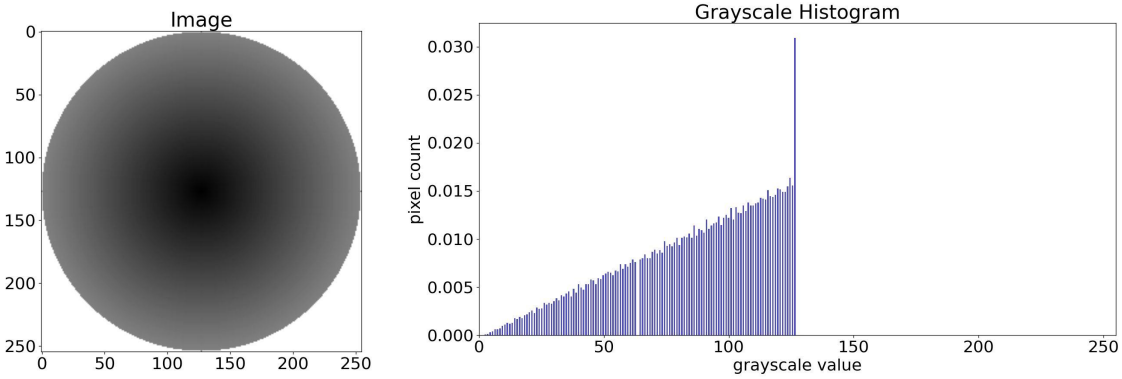
IC152: Assignment 8
Image Histograms and Binary Search

You must keep filenames as mentioned in the assignment since the assignments will be auto evaluated via a script. VERY IMPORTANT: Save all python files and outputs in a single folder. You must name python files as suggested in each question. Do not miss writing your roll number in the folder name of the zip file. Example folder name: b22000_assignment8.zip.

Problem 1: Image Creation and Image Histograms

a. Run problem1.py and try to understand it from comments.

b. Modify it to create the following figure and histogram (not bar at value 255 should be carefully removed from histogram). Save a python file with name problem1b.py and a common image having both subfigures with name problem1b.png.



Do not worry about the height of the last bar (shown above) at value near int(255/2). Only requirement for the histogram is that it should be increasing in range [0, int(255/2)).

c. Now read the `'problem1cInput.jpg'` image using the "helper code" given below and create the histogram of:
   i. Its pixel values
   ii. difference between each pixel with a pixel left to it (loop for column number from 1 to end). What do you observe? Share with the TA/instructor.

helper code starts:

```
# install pillow using:
# pip3 install pillow
# if above does not work and you are on windows, try:-
# py -m pip3 install pillow

# importing Image and ImageOps from PIL
from PIL import Image, ImageOps

# creating image object:
imInp = Image.open('problem1cInput.jpg')

# converting imInp to grayscale:
imGrayscale = ImageOps.grayscale(imInp)

# converting image to numpy array:
```

```
import numpy as np
pixels2D = np.array(imGrayscale)
print(pixels2D.shape)

# showing/saving image
imGrayscale.show()
#imGrayscale.save('problem1cOutput.jpg')
```
helper code ends.

Save your python file as problem1c.py, and running it should save two images with names problem1ci.png and problem1cii.png.

Problem 2: Binary Search

If you have a sorted array/list, you can apply binary search to quickly find a value's index in the array/list by i) comparing the middle value of the array with the value being searched and ii) sub arrays on its right or left depending on the middle value.

- E.g. if you want to search 15 in [3, 5, 7, 9, 11, 13, 15],
- Then you will compare 15 with the middle value i.e. 9.
- Now since 9 is less than 15, the next search will be in values right to 9, i.e. among the last three elements. (If we would have got the value in the middle as greater than 15 we would have continued searching in the left part, for the middle value equal to 15 we would have stopped).
- Now the middle value of the right part to 9 ([11, 13, 15]) is 13, which is less than 15. So we will search in the right part again.

- Now the middle value of part right to 13 ([15]) is 15 as only one element is left, and it is the equal to the value being searched. So, now we know the location of 15 and we stop our search.
- Observe that in 3 comparisons we have reached the last value. If we would have used a loop from 1 to n (7 in this case), we would have to do 7 comparisons. Hence binary search is the fastest way to search the index of a value in a sorted array.

a. Write a function to apply binary search on any generic input list and save it as problem2.py. The function should return the index of the value in the input array. The code should be named as problem2a.py and it should take filename with command line argument. The input file should have following content (for example) with first line having search value and second line having the array:
   20
   1 8 20 100 139 212 302 53113 1211

b. Write a code using a for loop. The code should be named as problem2b.py and it should take filename with command line argument.

c. Estimate the time problem2a.py and problem2b.py take for different sorted lists with around 100000 elements and share

your observation with TAs/instructor.

Note: Time can be estimated by importing time library as follows:

```
import time start = time.time()
#execution of the function
end = time.time()
total_time = (start-end)
```

Create the folder having your python files and inputs/outputs, with name having your roll number followed by "_assignment8" (don't use inverted commas in folder name), compress the folder with .zip extension and submit it on moodle.

Important: If you copy the assignment or any of its parts from others or share with others, our plagiarism softwares will catch it and you will be awarded 0 marks for the whole assignment or F grade for the course.