***Question 1:***

a)

*I think higher megapixels offer the advantage of larger prints without compromising quality. This becomes crucial for photography enthusiasts or professionals who may require expansive.*

*I think the camera's processing capabilities are pivotal. Higher megapixels mean larger image files, and it requires swift processing, especially in scenarios like continuous shooting or video recording.*

*Adding to that rendering detailed images it can be computationally intensive. Higher resolution requires more processing time and affecting the efficiency of workflows, particularly in fields like animation and film production.*

*Moreover higher resolution images come at the cost of more storage space. The decision on resolution, I think, involves a delicate trade-off between image quality and storage considerations, especially where space is limited.*

*Generating high-resolution images, I think, demands robust processing power. The GPU's capability plays a crucial role, particularly in real-time applications like gaming or video editing.*

b)

in my opinion, factor into the decision on how strong the quantization of an image should be:

The size of the image: A larger image will have more pixels and require more storage space and bandwidth. Quantizing the image can reduce the size and make it easier to transmit and store.

Also Different hardware devices have different capabilities and limitations. For example, older computer hardware or less advanced screens may not support a high number of colors or resolution, while newer devices may have more advanced features and specifications. Quantizing the image can affect the compatibility and performance of the image on different hardware devices and it can show it on the screen better without allowing the color loss make the image hard to see.

Also Recognizing the limitations of the human eye in perceiving subtle color differences, I think quantization can be adjusted to match the eye's sensitivity, optimizing overall visual perception.

**Question 2:**

a) To find the wavelength for this image (sin) along the x-axis we will need to determine the distance between the two continuous lines for the sin function

$the\ \sin(\alpha)\ function\ make\ a\ full\ cycle\ \ when\ \alpha$ gets bigger by $2\pi$

$2\pi = \pi kx \rightarrow x = \frac{2}{k} = \lambda$

$I(x, y) = \sin(\ \pi kx)$ so in this case the wavelength $\lambda$ along the x-axis will be:

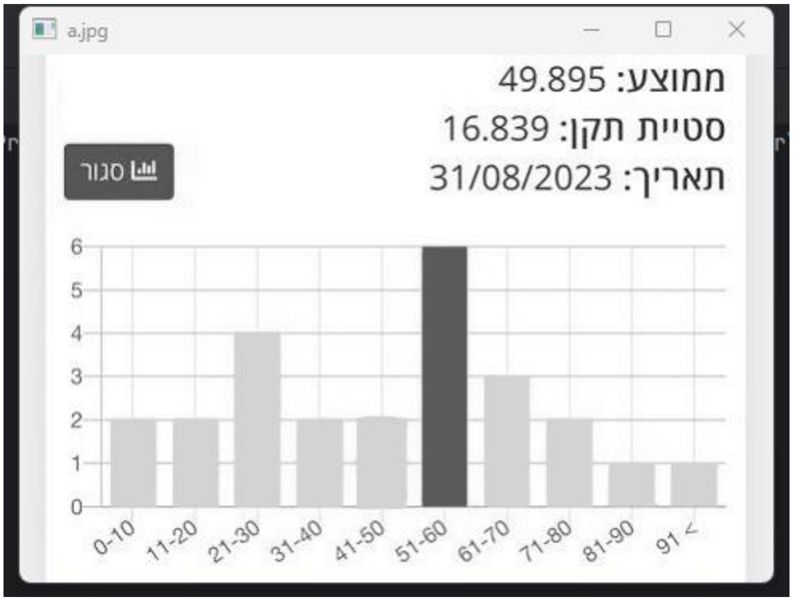
$\lambda = \frac{2\pi}{\pi k} = \frac{2}{k} \quad f = \frac{1}{\lambda} = \frac{k}{2}$

b) From the previous answer we get that
$f = \frac{1}{\lambda} = \frac{k}{2}$ so, from NYQUIST law if we want to get back the picture precisely without any errors, we will need to get that $f_{sample} \geq 2f$
For A=2:
The wavelength is 4 cm (because of the distance between two lines is 2A == wavelength)
Also we get that the $f_{sample} = \frac{1}{4} = \frac{1}{\lambda}$ So ➔ from NYQUIST:
$f_{sample} \geq 2f \quad \rightarrow \quad \frac{1}{4} \geq 2\frac{k}{2} \quad \rightarrow \quad \frac{1}{4} \geq k$

For A=0.25:
The same as before we get that the wavelength is 0.5 cm
Also we get that the $f_{sample} = \frac{1}{0.5} = 2 = \frac{1}{\lambda}$ So ➔ from NYQUIST:
$f_{sample} \geq 2f \quad \rightarrow \quad 2 \geq 2\frac{k}{2} \quad \rightarrow \quad 2 \geq k$

**Question 3:**

**a)** Reading the images and display the first one using cv2



ממוצע: 49.895
סטיית תקן: 16.839
תאריך: 31/08/2023

b) numbers, _ = read_dir('numbers')

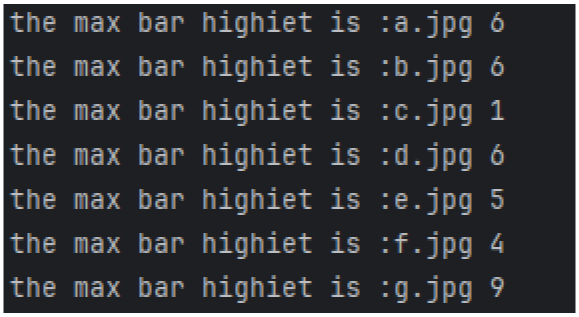
c)

This how I found the wanted window we wanted to check the target in:

1. I choose some image that the max par Hight of it is 6
2. Then I go through all the windows and checked if the emd <260
3. If yes I will print the x , y of the window that it return to me true
4. Because we know that all the images from the same format we can
   Now when we do compare_hist just check the emd for window[x , y] and the target

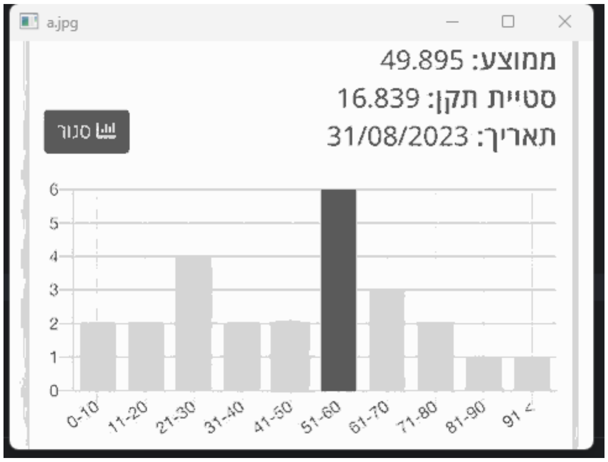
In our case it's [113,30]

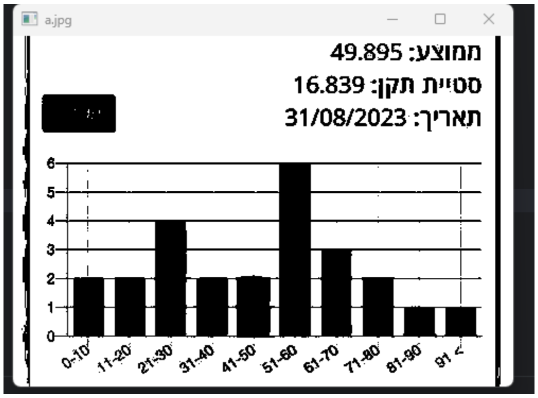
d) This is the max- bar height that we get from applying the compare hist in all the images:



```
the max bar highiet is :a.jpg 6
the max bar highiet is :b.jpg 6
the max bar highiet is :c.jpg 1
the max bar highiet is :d.jpg 6
the max bar highiet is :e.jpg 5
the max bar highiet is :f.jpg 4
the max bar highiet is :g.jpg 9
```

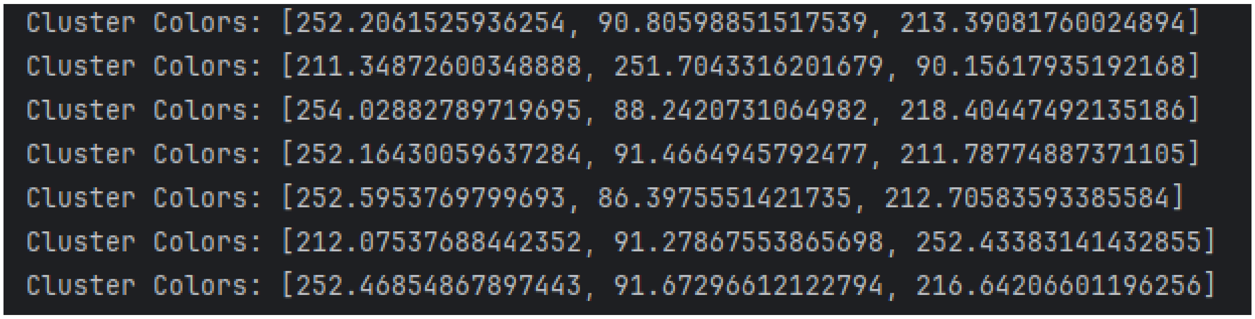
e)





I choose n_color=3 because it's the minimal n_color number that I can do it safely with (without loosing the bars) with two I will not be able to see the difference between the bars and the background

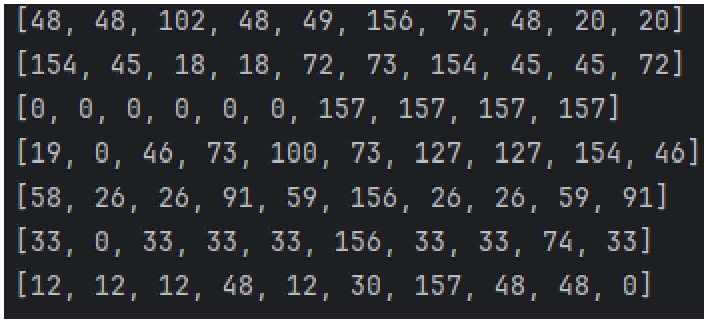
Also I choose thresh of 245 because I printed colors that the cluster makes for each image and it shows that all the pars get ~250



```
Cluster Colors: [252.2061525936254, 90.80598851517539, 213.39081760024894]
Cluster Colors: [211.34872600348888, 251.7043316201679, 90.15617935192168]
Cluster Colors: [254.02882789719695, 88.2420731064982, 218.40447492135186]
Cluster Colors: [252.16430059637284, 91.4664945792477, 211.78774887371105]
Cluster Colors: [252.5953769799693, 86.3975551421735, 212.70583593385584]
Cluster Colors: [212.07537688442352, 91.27867553865698, 252.43383141432855]
Cluster Colors: [252.46854867897443, 91.67296612122794, 216.64206601196256]
```

So by putting the thresh on 245 it certainly we include all the bars in the black (color value 0 )

f) The bar heights for all the images:



```
[48, 48, 102, 48, 49, 156, 75, 48, 20, 20]
[154, 45, 18, 18, 72, 73, 154, 45, 45, 72]
[0, 0, 0, 0, 0, 0, 157, 157, 157, 157]
[19, 0, 46, 73, 100, 73, 127, 127, 154, 46]
[58, 26, 26, 91, 59, 156, 26, 26, 59, 91]
[33, 0, 33, 33, 33, 156, 33, 33, 74, 33]
[12, 12, 12, 48, 12, 30, 157, 48, 48, 0]
```

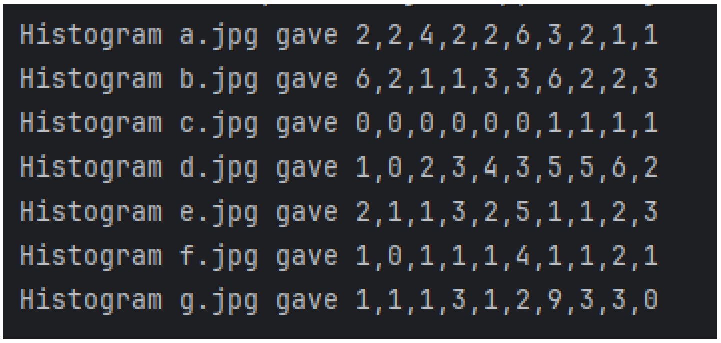
In this function I changes the implementation of it to suit the the white_and_black images

So in my case the bars in black with a color value 0 so I changes the while to check if the color 0

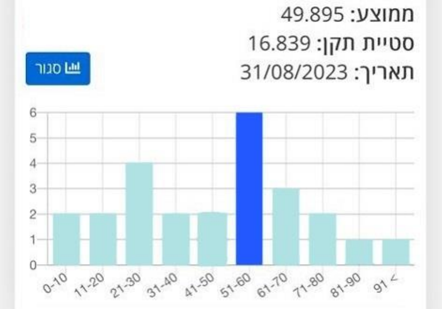
In get_bar_height function

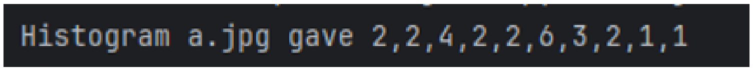
g) Form all the info we get before now we will calculate the students-per-bin = round(max-student-num * bin-height / max-bin-height) for each bar using this
max-student-num = we get in from compare_hist (doing for loop from 9 to 0 )
bin-height = we get in from the highest we gen from the (f)
finally the max-bin-height = we also get it from (f) but now we will take the maximum height in pixels it's always the same for all the image as we can we the max bar height in all the images in average is ~154 so I used it (max-bin-height is the height of the bar when I reaches the maximum student number)

The final output for the given data:

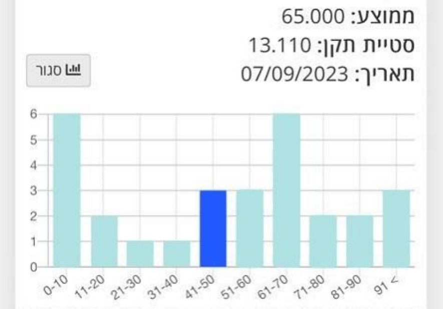


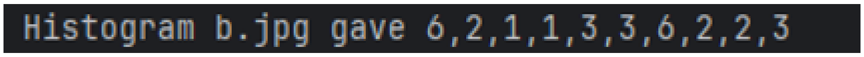
```
Histogram a.jpg gave 2,2,4,2,2,6,3,2,1,1
Histogram b.jpg gave 6,2,1,1,3,3,6,2,2,3
Histogram c.jpg gave 0,0,0,0,0,0,1,1,1,1
Histogram d.jpg gave 1,0,2,3,4,3,5,5,6,2
Histogram e.jpg gave 2,1,1,3,2,5,1,1,2,3
Histogram f.jpg gave 1,0,1,1,1,4,1,1,2,1
Histogram g.jpg gave 1,1,1,3,1,2,9,3,3,0
```



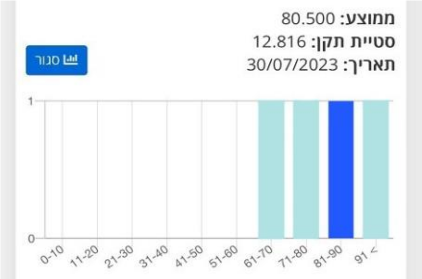


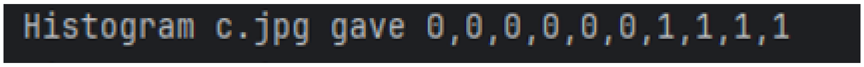
```
Histogram a.jpg gave 2,2,4,2,2,6,3,2,1,1
```



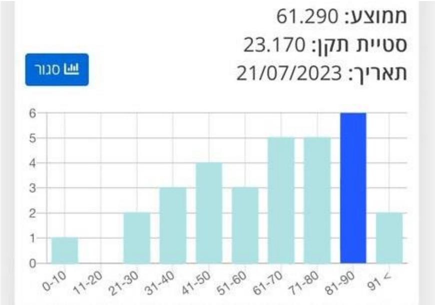


```
Histogram b.jpg gave 6,2,1,1,3,3,6,2,2,3
```

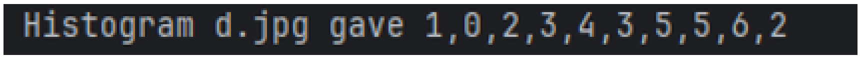




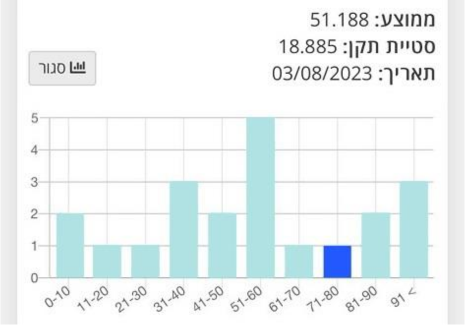
```
Histogram c.jpg gave 0,0,0,0,0,0,1,1,1,1
```



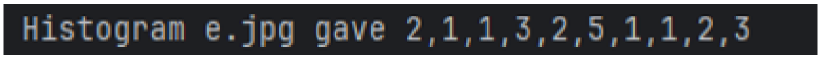
ממוצע: 61.290
סטיית תקן: 23.170
תאריך: 21/07/2023



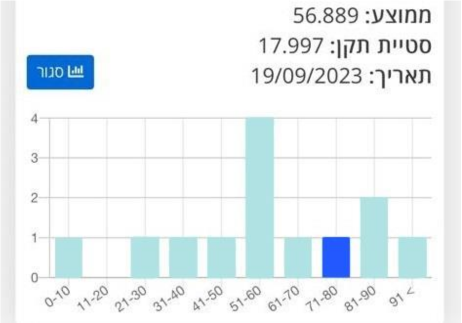
```
Histogram d.jpg gave 1,0,2,3,4,3,5,5,6,2
```



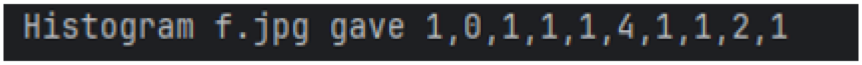
ממוצע: 51.188
סטיית תקן: 18.885
תאריך: 03/08/2023



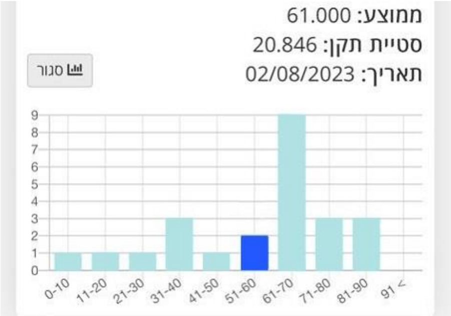
```
Histogram e.jpg gave 2,1,1,3,2,5,1,1,2,3
```



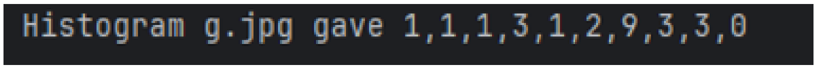
ממוצע: 56.889
סטיית תקן: 17.997
תאריך: 19/09/2023



```
Histogram f.jpg gave 1,0,1,1,1,4,1,1,2,1
```



ממוצע: 61.000
סטיית תקן: 20.846
תאריך: 02/08/2023



```
Histogram g.jpg gave 1,1,1,3,1,2,9,3,3,0
```