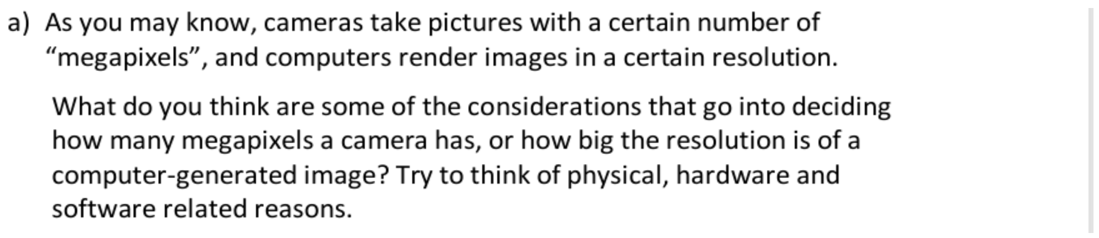
IMAGE PROCCESSING REPORT _ HW1

lana shihab 206539199

Ahmad Igbaria 322751041

Q1:



a) As you may know, cameras take pictures with a certain number of "megapixels", and computers render images in a certain resolution.

What do you think are some of the considerations that go into deciding how many megapixels a camera has, or how big the resolution is of a computer-generated image? Try to think of physical, hardware and software related reasons.

Image Editing :

High resolution images porivde more flexibility during the editing process such as cropping , color adjustments and fine detail manipulation without compromising image quality.

Storage:

Higher resolion images require more storage space on devices .

The number of megapixels dicretly affects the file size which requires more memory on memory cars and computers

Processing Power :

Heiher resolion images require more computational power to process .

Computers need stronger resources to work with large images , including more RAM and a powerful graphics card.

Workflow :

Working with high resolution images allows for more precies cuts and better results,especially in professional fields but requires and advanced computiong system to maintaion smooth workflow .

Long Term Quality Preservation:

High resolution images maintaion high quality even when printed large or used again without losing details or pixelation

Viewwing Experience:

High resolution images provide a sharper and clearer viewing experience especially on larger displays like 4k or 8k screens or large prints .

b) After an image is sampled, it needs to be quantized before we can actually see it in its final form (remember that computers are mostly digital devices, so analogue values can’t just be stored “as-is” in memory).

What do you think are some of the considerations that go into deciding how strong the quantization of an image is? Try to think back about older computer hardware or less advanced screens.

Transmission speed :

Stronger quantization reduces file size , allwing faster transmission over networks with limited bandwidth .

Thus is crucial for older systems with slower network connections.

Memrory :

Older computers with limited RAM can’t handle high resolution images.

Strong quanztion reduces the memory needed , making thE image suitable for older systems

Display compatibility with older screens:

Older screens were limited in color range , so stronger quantitizion was needed to isplay images clearly . this often resulted in color banding and a lack of smooth gradients , redcing overall image quality.

File Size :

Quantization reduces the number of bits per pixel , which decreases the file size.

This is especially important for older computers with limited storage which struggle to store large files .

Viewing experience compared to modern screens:

Today newer screens support weaker quantization , allwing clearer and more vibrant image display .

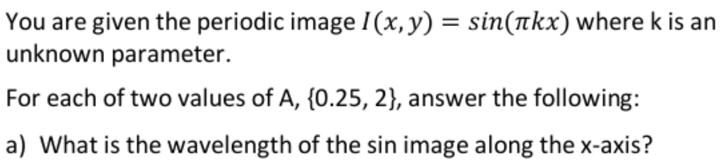
This enables smoother color transition and more detailed images , enhancing the overall viewing experience .

Color accuracy :

Older screens required stronger quantization , leading to a loss of color accuracy .

Today with advanced screens weaker quanztizaition can be used , preserving better color accuracay .

שאלה 2)


You are given the periodic image $I(x,y) = sin(\pi kx)$ where k is an unknown parameter.

For each of two values of A, {0.25, 2}, answer the following:

a) What is the wavelength of the sin image along the x-axis?

סעיף א)

נעשה משהו דומה כמו שעשינו בתרגול 1 ,

נתונה התמונה המחזורית $I(x,y) = \sin(\pi kx)$, קודם כל נתחיל להבין איך זו עובדת

באופן כללי פונקציה סינסוית מתאפיינת $\sin\,(2\pi k)$ במחזור של אורך $\frac{1}{k}$ כלומר כל פעם ש x משתנה ב

$\frac{1}{k}$ הפונקציה מבצעת מחזר שלם , לכן עבור הפונקציה $sin(\pi kx)$ אורך המחזור הוא $\frac{2}{k}$

כאשר $k=1$: אורך הגל יהיה $\frac{2}{1} = 2$ הפונקציה מבצעת מחזור שלם כך ש x משתנה בין 0 ל 2

כאשר $k=2$: אורך הגל יהיה $\frac{2}{2} = 1$. כלומר הפונקציה תבצע מחזור שלם כך שX משתנה בין 0 עד 1

כאשר $k=3$: אור ך הגל יהיה $\frac{2}{3}$ כלומר הפונקציה תבצע מחזור שלם כאשר X משתנה מ 0 עד $\frac{2}{3}$

כאשר $k=4$: אור ך הגל יהיה $\frac{1}{2}$ כלומר הפונקציה תבצע מחזור שלם כאשר X משתנה מ 0 עד $\frac{1}{2}$

וכך הלא ... אנחנו רואים שעבור כל ערך K אורך הגל יהיה $\frac{2}{k}$

סעיף ב)

כדי לשחזר את התמונה הסינסוית בצורה מדויקת בעזרת רשת דגימה , עלינו להשתמש בתנאי משפט הדיגמה שאומר :

$$f_{sample} \geq 2f_1$$

אורך הגל של כל דגימה הוא $2A$ כי הדגימות ברשת הן העמודות השחורות ועלינו קודם להגיע לעמודות הלבנות ואז לחזור לשחורות ולכן אורך הגל $A + A = 2A$

תדירות הדגימה = 1/אורך הגל שלה לכן :

$$f_{sample} = \frac{1}{\lambda_{sample}} = \frac{1}{2A}$$

נזכור שנתון שגל הסינוס הנתון הוא $\sin(\pi kx)$ ולכן מסעיף א :

$$\frac{K}{2} = \frac{1}{\frac{2}{K}} = f_I$$

נשתמש עכשיו בתנאי משפט הדגימה , לכן צריך להתקיים :

$$f_{sample} \geq 2f_1$$

$$\frac{1}{2A} \geq \frac{2k}{2} = k$$

$$\rightarrow \frac{1}{2A} \geq k$$

נציב עכשיו *a=0.25* , *a=2* ונבדוק מה יוצא

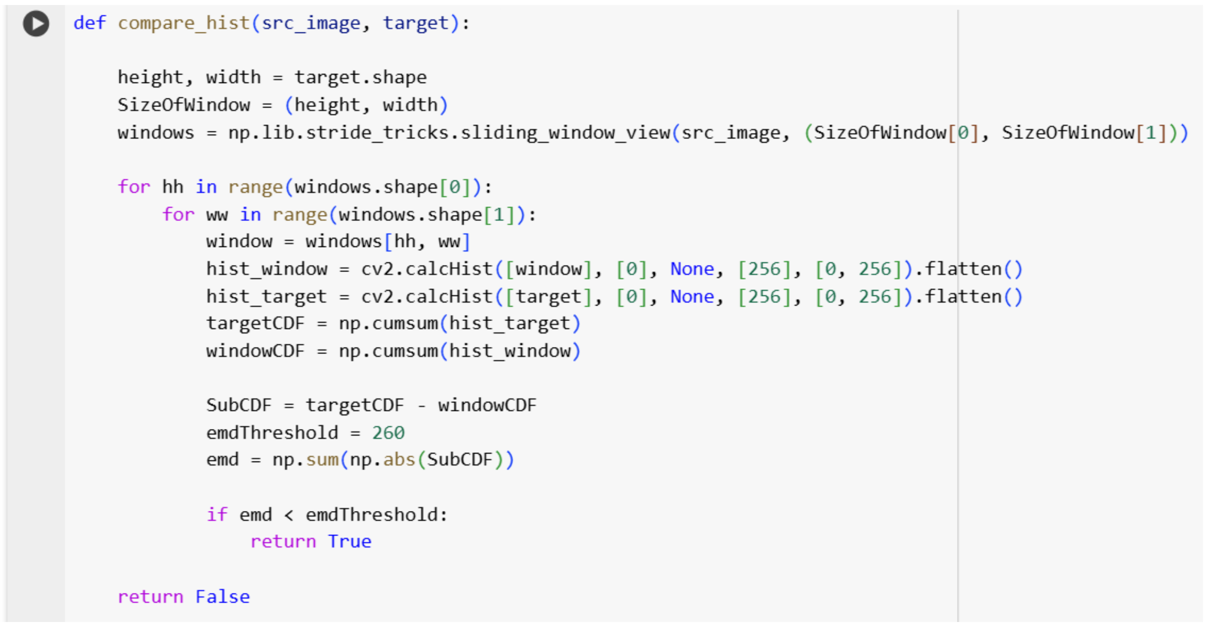
$$a = 2 \rightarrow \frac{1}{2 * 2} \geq k \ \rightarrow \frac{1}{4} \geq k$$

$$a = 0.25 \rightarrow \frac{1}{2 * 0.25} \geq k \rightarrow 2 \geq k$$

שאלה 3)

קודם כל אני אסביר קצת על כמה פונקציות ואז אני מראה הרבה תוצאות

Compare_hist Function

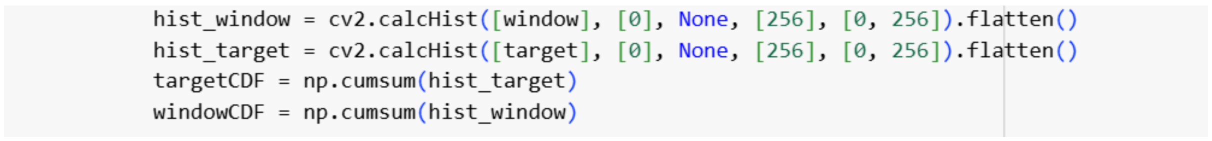


```python
def compare_hist(src_image, target):

    height, width = target.shape
    SizeOfWindow = (height, width)
    windows = np.lib.stride_tricks.sliding_window_view(src_image, (SizeOfWindow[0], SizeOfWindow[1]))

    for hh in range(windows.shape[0]):
        for ww in range(windows.shape[1]):
            window = windows[hh, ww]
            hist_window = cv2.calcHist([window], [0], None, [256], [0, 256]).flatten()
            hist_target = cv2.calcHist([target], [0], None, [256], [0, 256]).flatten()
            targetCDF = np.cumsum(hist_target)
            windowCDF = np.cumsum(hist_window)

            SubCDF = targetCDF - windowCDF
            emdThreshold = 260
            emd = np.sum(np.abs(SubCDF))

            if emd < emdThreshold:
                return True

    return False
```

this function designed to compare histograms between a soucre imag and a target image .

it purpose is to detemine wether a specific region in the soucre image matches the target image , based on EMD between their histograms

Caclationg Histograms by calcHist (cv2)



```python
            hist_window = cv2.calcHist([window], [0], None, [256], [0, 256]).flatten()
            hist_target = cv2.calcHist([target], [0], None, [256], [0, 256]).flatten()
            targetCDF = np.cumsum(hist_target)
            windowCDF = np.cumsum(hist_window)
```

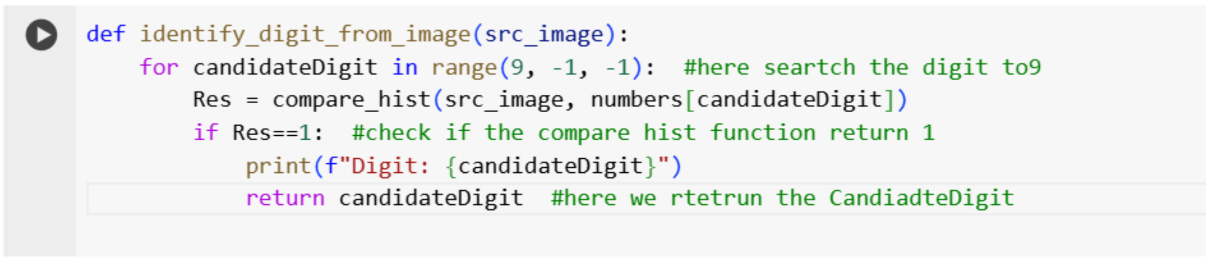
how this function works ?

creating fixed size sliding windows from the source image , based on the size of the target image

calclatuing histograms for each window and for the target image

We used the CalcHist function as Alon requseted to compute the histogram

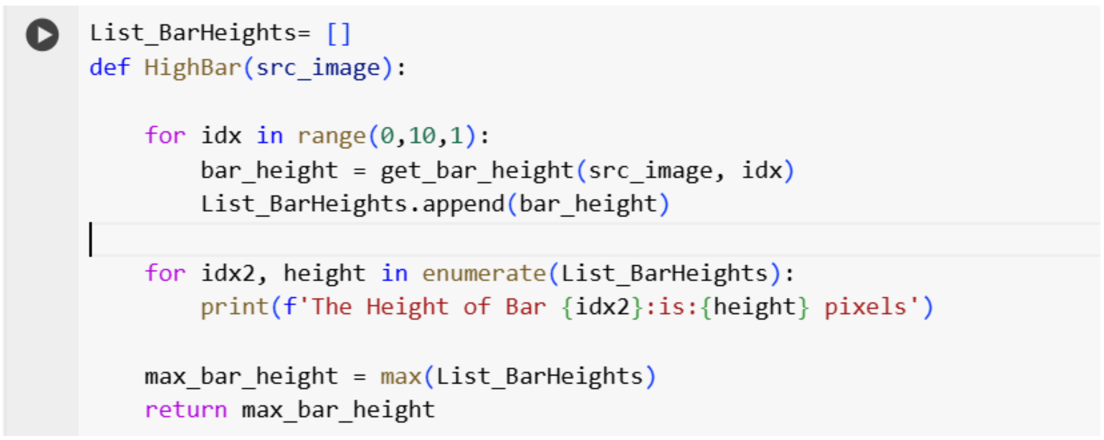
Employing the EMD to mesure the similarity between the histogram distributions



```python
def identify_digit_from_image(src_image):
    for candidateDigit in range(9, -1, -1):  #here seartch the digit to9
        Res = compare_hist(src_image, numbers[candidateDigit])
        if Res==1:  #check if the compare hist function return 1
            print(f"Digit: {candidateDigit}")
            return candidateDigit  #here we rtetrun the CandiadteDigit
```

the identify_digit_from_image function identifies a digit in an image by comparing the images histogram to digits from 9 to 0 . it uses the Compare_hist function to check if there is a match between the reqion in the image and one of the digits . if a match is found (compare hist function returns 1) the function prints and returuns the corresponding digit .

the purpose is to identify the highest digit presnt in the image

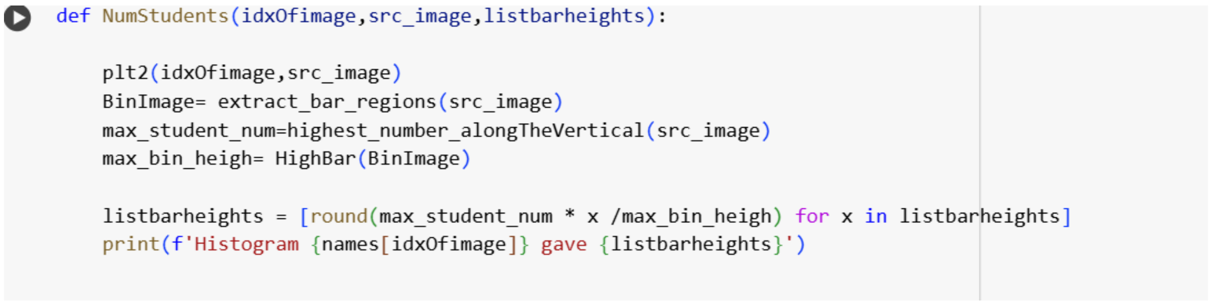


```python
List_BarHeights= []
def HighBar(src_image):

    for idx in range(0,10,1):
        bar_height = get_bar_height(src_image, idx)
        List_BarHeights.append(bar_height)

    for idx2, height in enumerate(List_BarHeights):
        print(f'The Height of Bar {idx2}:is:{height} pixels')

    max_bar_height = max(List_BarHeights)
    return max_bar_height
```

the HighBar function calculates the heights of 10 bars in an image and returns the maximum height . it uses the get_bar_height function to measure to height of each bar , stores the heights in the List_BarHeights list, and prints the height of each bar .

Finally , it finds the maximim height using max and returns it .

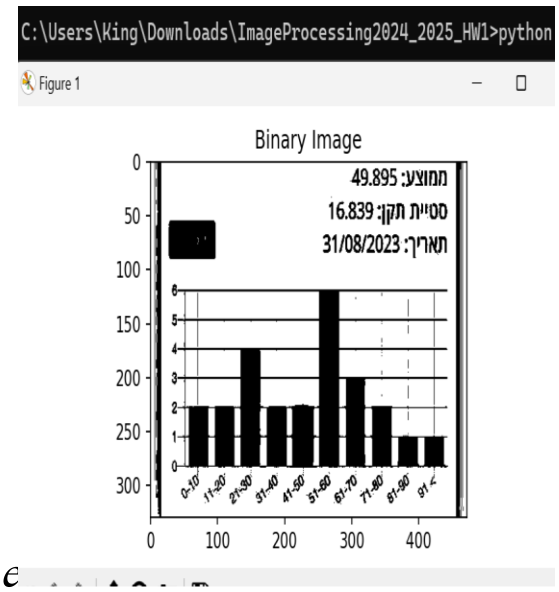
the purpose is to analyze the bar heights for furhter calculations such as normaliztion or determinig the number of students



```
def NumStudents(idxOfimage,src_image,listbarheights):

    plt2(idxOfimage,src_image)
    BinImage= extract_bar_regions(src_image)
    max_student_num=highest_number_alongTheVertical(src_image)
    max_bin_heigh= HighBar(BinImage)

    listbarheights = [round(max_student_num * x /max_bin_heigh) for x in listbarheights]
    print(f'Histogram {names[idxOfimage]} gave {listbarheights}')
```

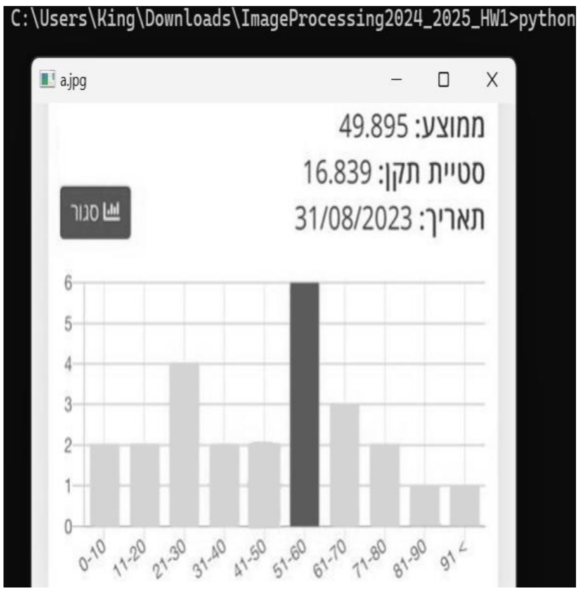
the NumStudents function calculates the number os students for each bar in the image based on the heights of the bars obtained from the histograms it uses different functions to extract the bar heights from the image (extract_bar_regions and HighBar) and compares the obtained heights to the maximum number of students . Finally , it prints the number of studnets for each bar in the image

now For each image, I display the histogram, the output, and the binary image *first image :*

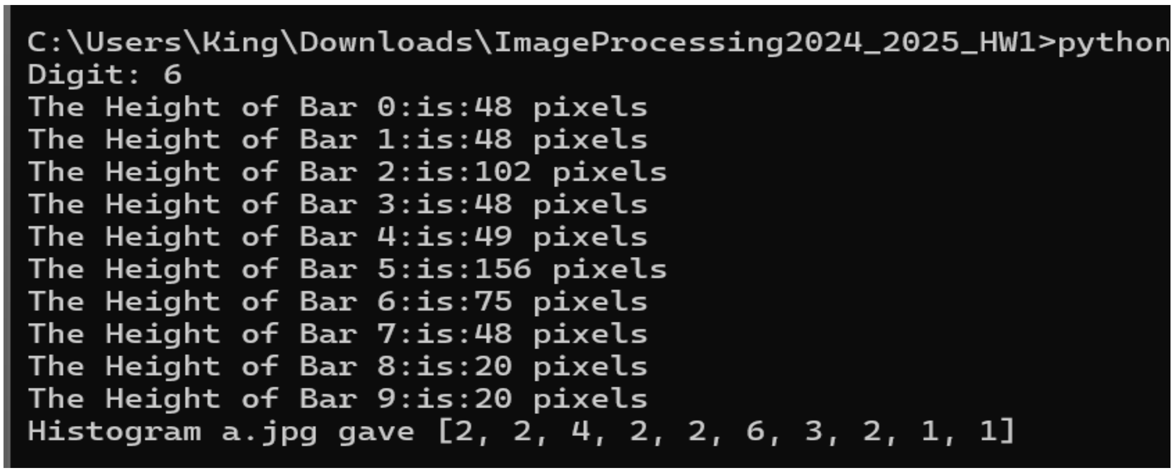


```
C:\Users\King\Downloads\ImageProcessing2024_2025_HW1>python
```

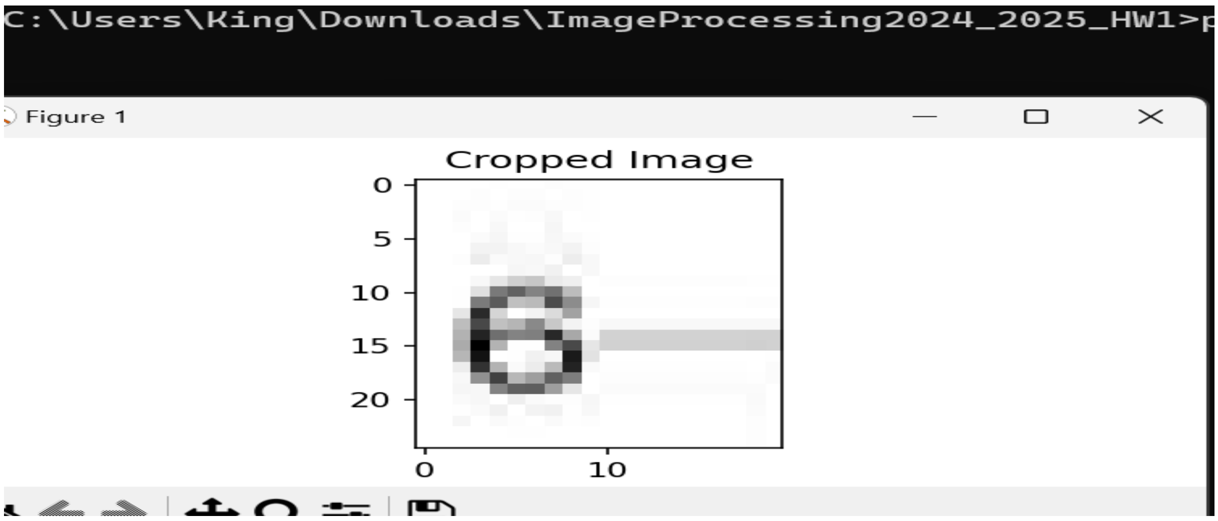
e :



```
C:\Users\King\Downloads\ImageProcessing2024_2025_HW1>python
```

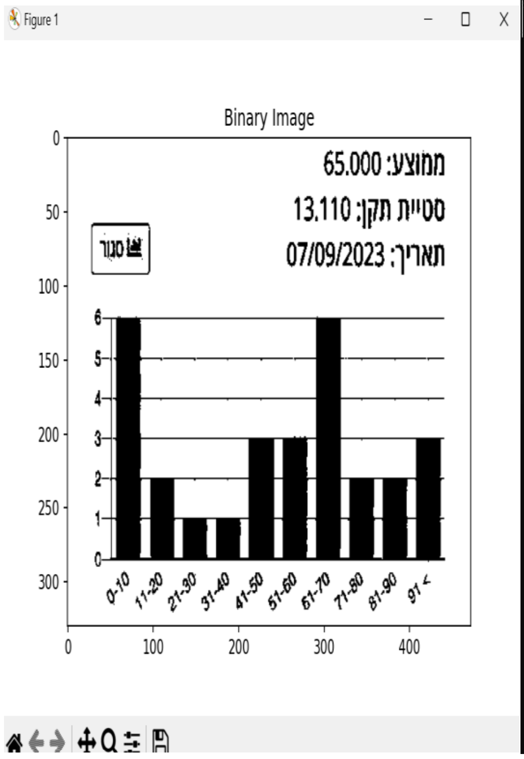


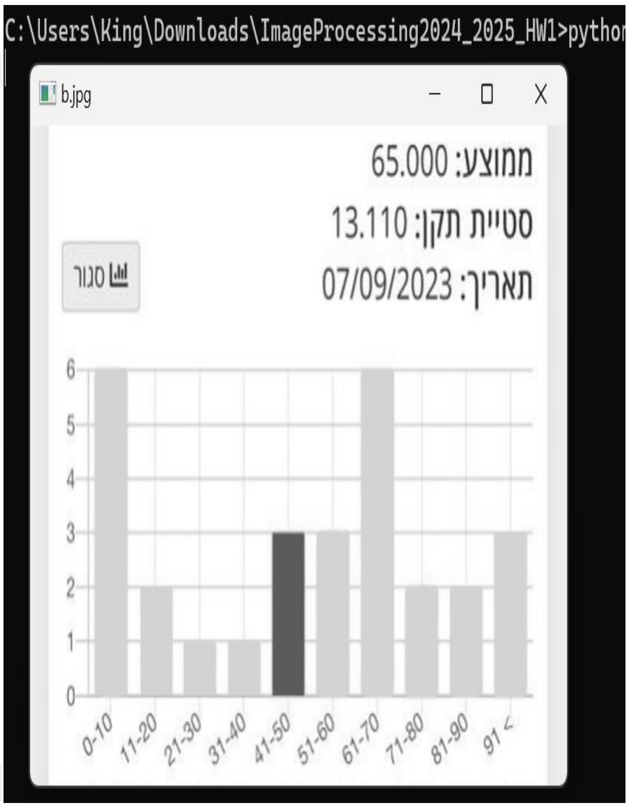
```
C:\Users\King\Downloads\ImageProcessing2024_2025_HW1>python
Digit: 6
The Height of Bar 0:is:48 pixels
The Height of Bar 1:is:48 pixels
The Height of Bar 2:is:102 pixels
The Height of Bar 3:is:48 pixels
The Height of Bar 4:is:49 pixels
The Height of Bar 5:is:156 pixels
The Height of Bar 6:is:75 pixels
The Height of Bar 7:is:48 pixels
The Height of Bar 8:is:20 pixels
The Height of Bar 9:is:20 pixels
Histogram a.jpg gave [2, 2, 4, 2, 2, 6, 3, 2, 1, 1]
```



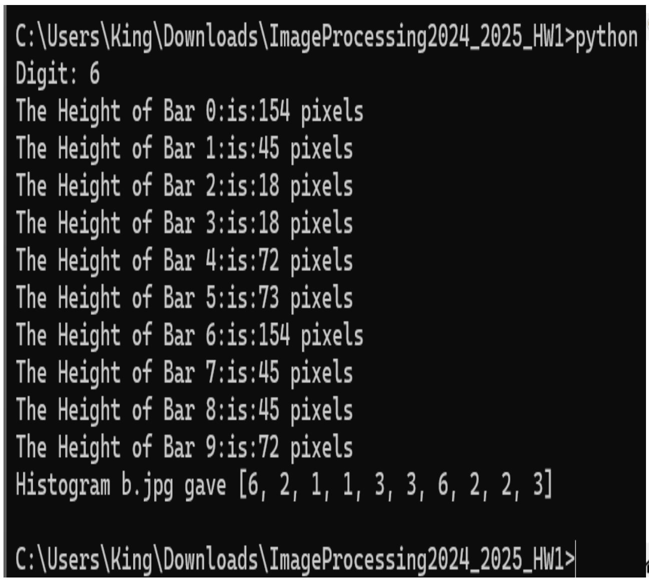
```
C:\Users\King\Downloads\ImageProcessing2024_2025_HW1>p
```

Second image :

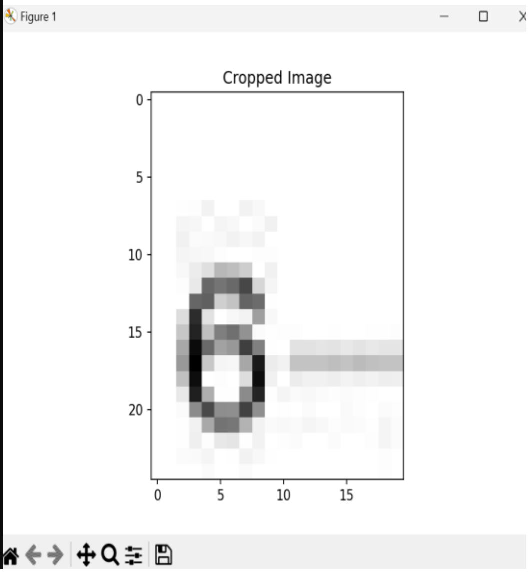




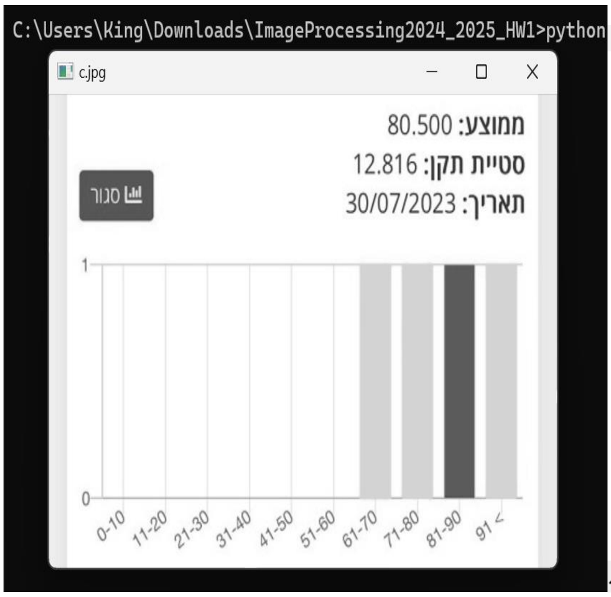
```
C:\Users\King\Downloads\ImageProcessing2024_2025_HW1>pytho
```



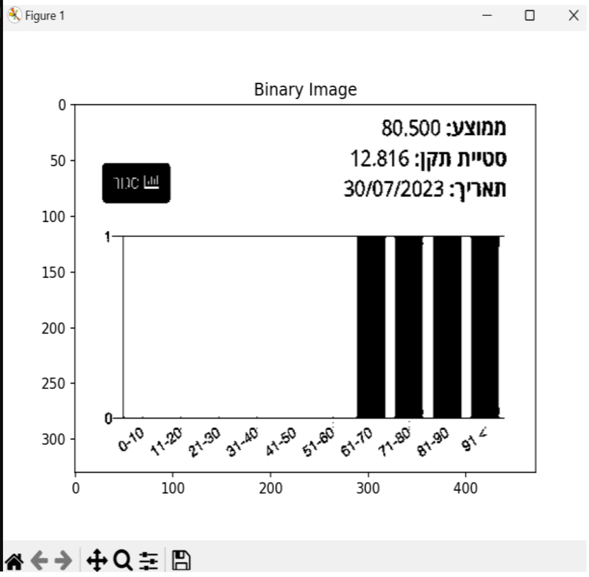
```
C:\Users\King\Downloads\ImageProcessing2024_2025_HW1>python
Digit: 6
The Height of Bar 0:is:154 pixels
The Height of Bar 1:is:45 pixels
The Height of Bar 2:is:18 pixels
The Height of Bar 3:is:18 pixels
The Height of Bar 4:is:72 pixels
The Height of Bar 5:is:73 pixels
The Height of Bar 6:is:154 pixels
The Height of Bar 7:is:45 pixels
The Height of Bar 8:is:45 pixels
The Height of Bar 9:is:72 pixels
Histogram b.jpg gave [6, 2, 1, 1, 3, 3, 6, 2, 2, 3]

C:\Users\King\Downloads\ImageProcessing2024_2025_HW1>
```

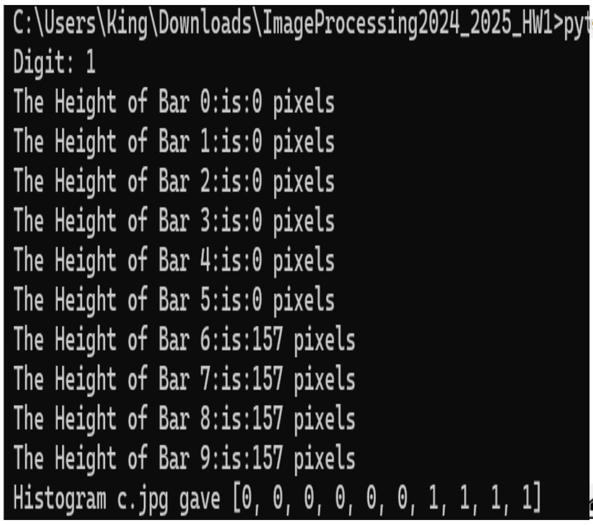


Third image :

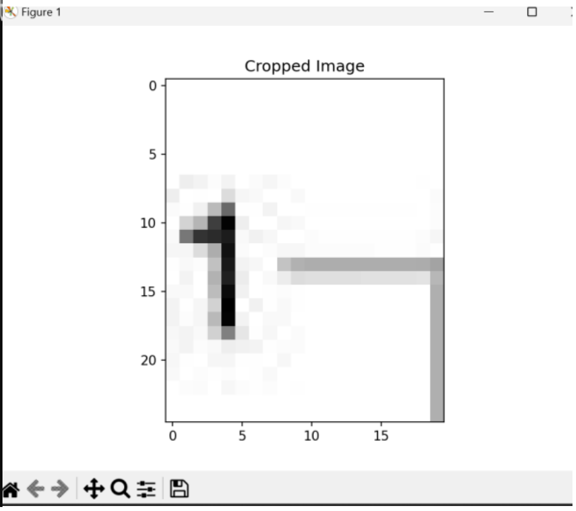


```
C:\Users\King\Downloads\ImageProcessing2024_2025_HW1>python
```

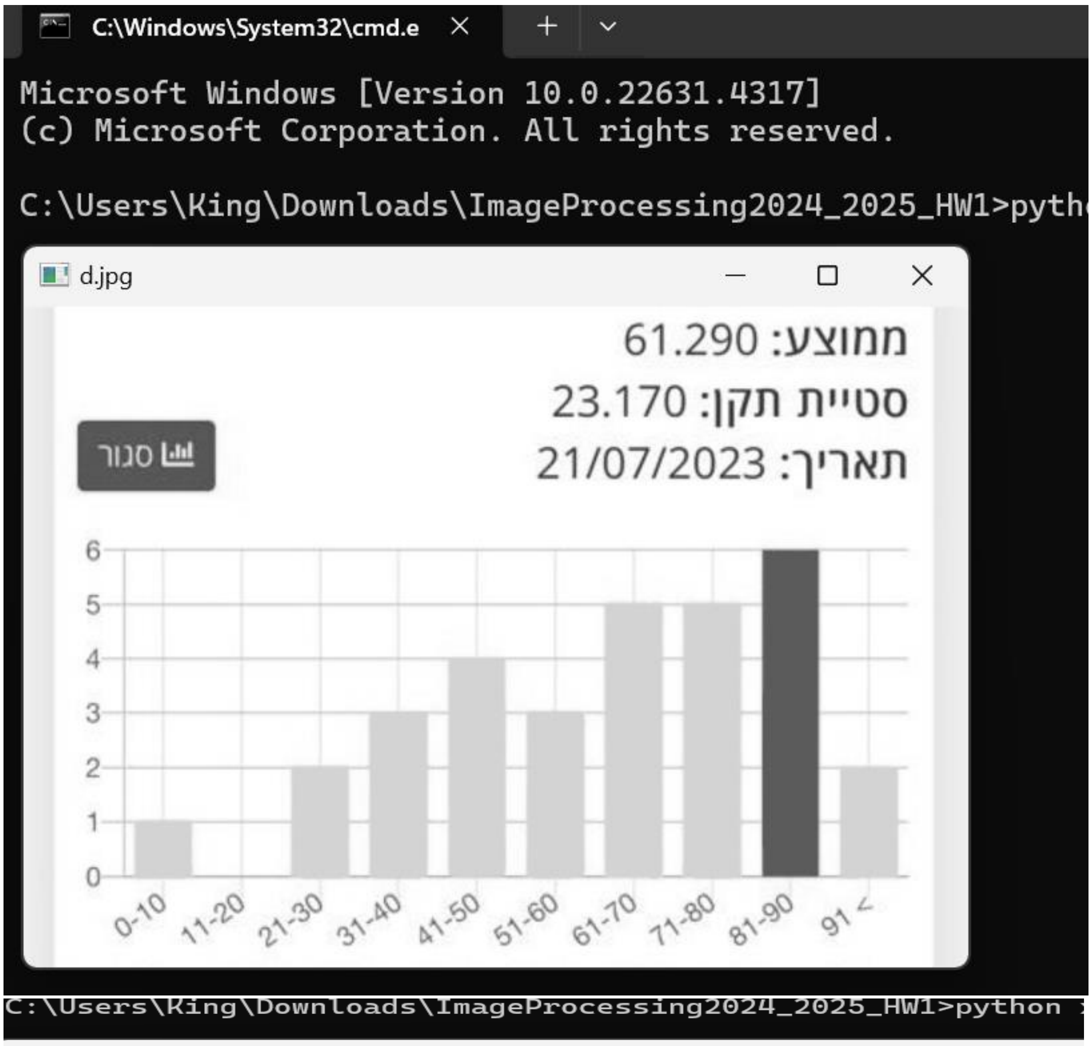




```
C:\Users\King\Downloads\ImageProcessing2024_2025_HW1>py
Digit: 1
The Height of Bar 0:is:0 pixels
The Height of Bar 1:is:0 pixels
The Height of Bar 2:is:0 pixels
The Height of Bar 3:is:0 pixels
The Height of Bar 4:is:0 pixels
The Height of Bar 5:is:0 pixels
The Height of Bar 6:is:157 pixels
The Height of Bar 7:is:157 pixels
The Height of Bar 8:is:157 pixels
The Height of Bar 9:is:157 pixels
Histogram c.jpg gave [0, 0, 0, 0, 0, 0, 1, 1, 1, 1]
```

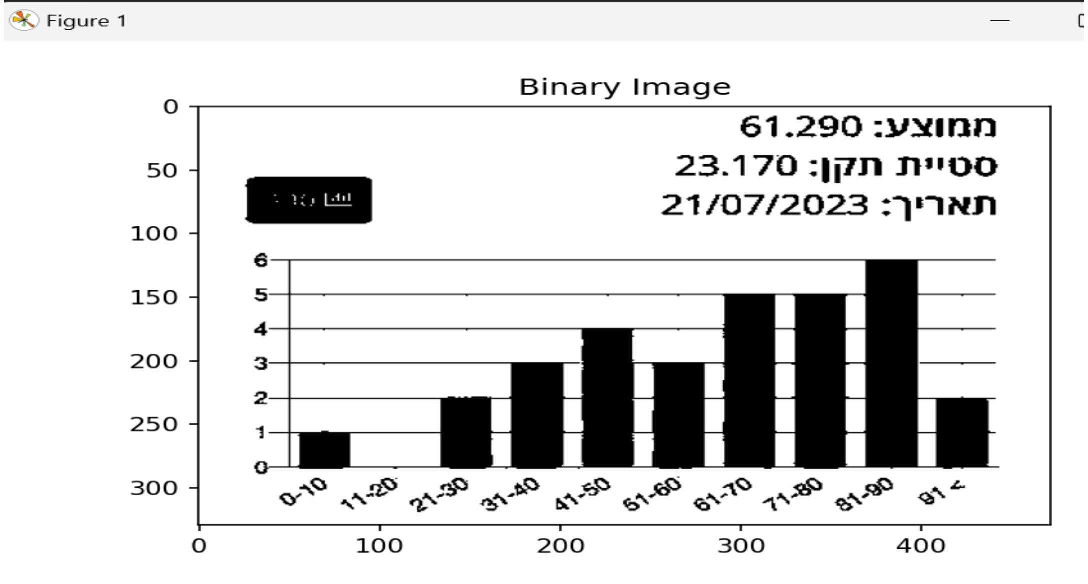


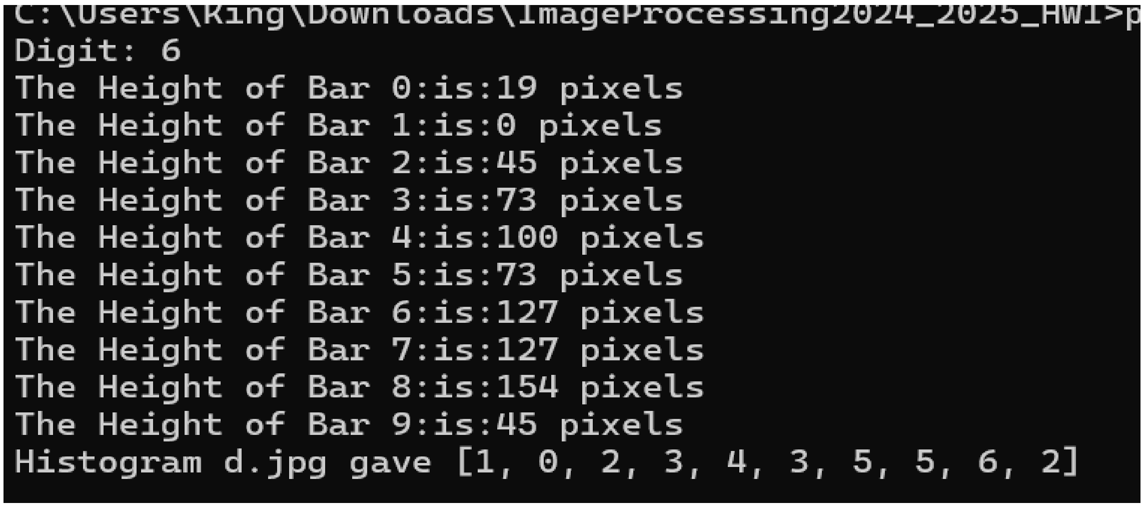
fourth image :



```
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\King\Downloads\ImageProcessing2024_2025_HW1>pyth
```

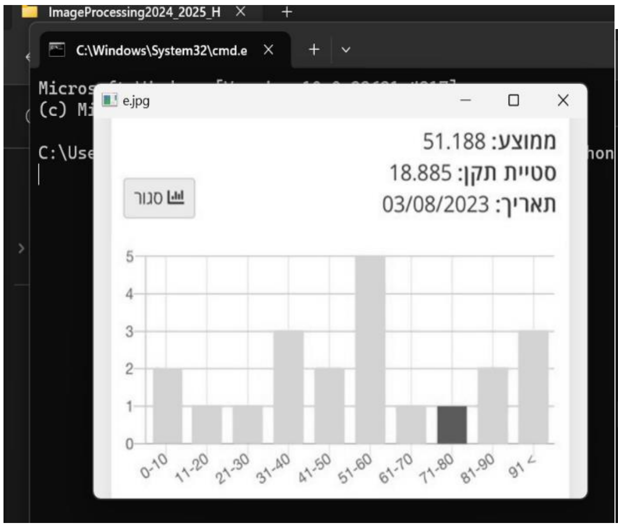
```
C:\Users\King\Downloads\ImageProcessing2024_2025_HW1>python
```

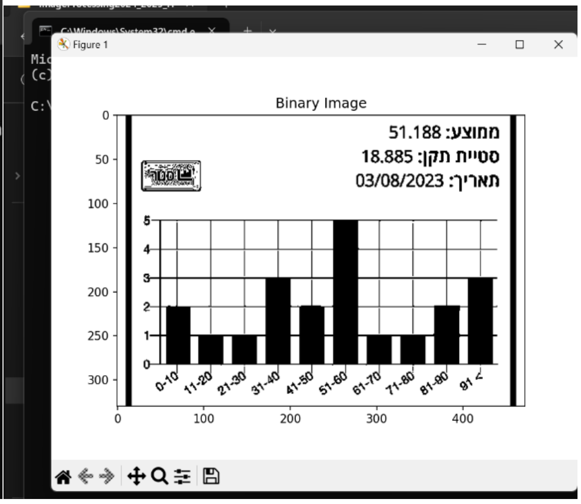


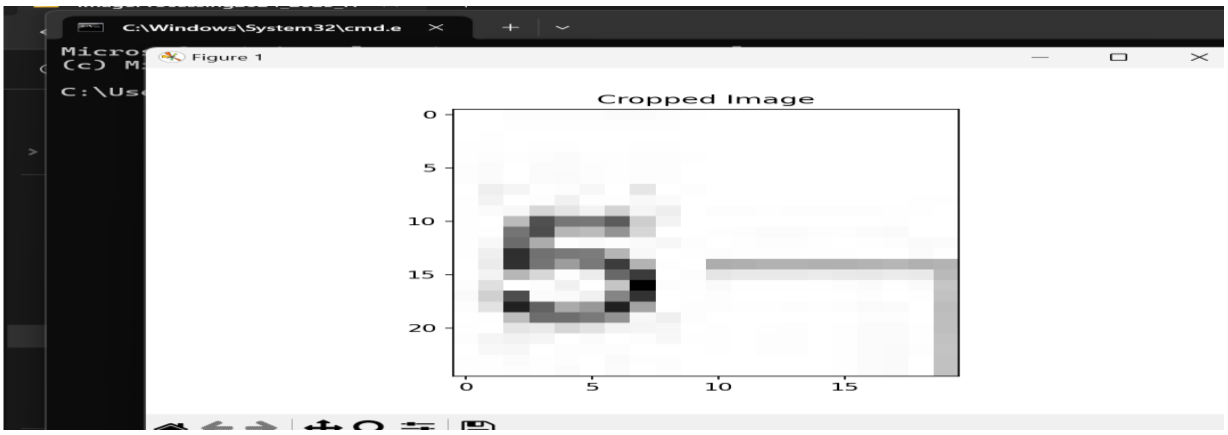


```
C:\Users\King\Downloads\ImageProcessing2024_2025_HW1>p
Digit:  6
The Height of Bar 0:is:19 pixels
The Height of Bar 1:is:0 pixels
The Height of Bar 2:is:45 pixels
The Height of Bar 3:is:73 pixels
The Height of Bar 4:is:100 pixels
The Height of Bar 5:is:73 pixels
The Height of Bar 6:is:127 pixels
The Height of Bar 7:is:127 pixels
The Height of Bar 8:is:154 pixels
The Height of Bar 9:is:45 pixels
Histogram d.jpg gave [1, 0, 2, 3, 4, 3, 5, 5, 6, 2]
```

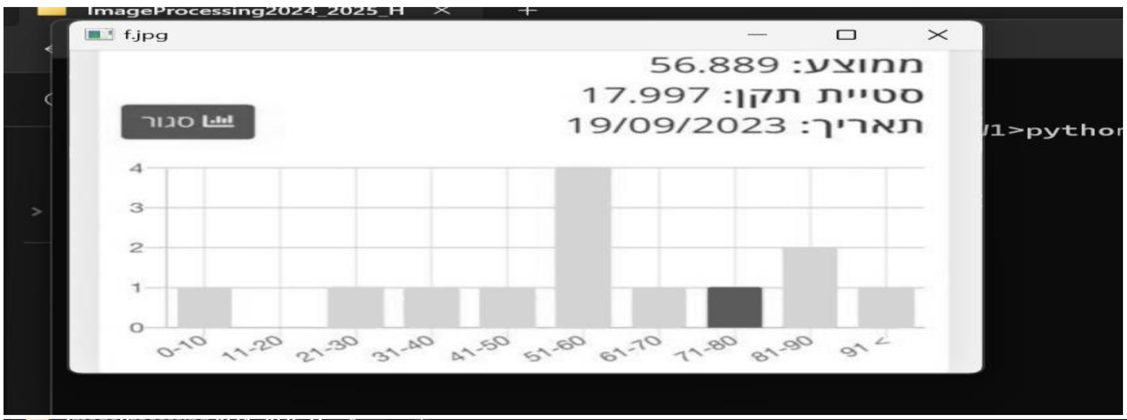
Fiveth image:

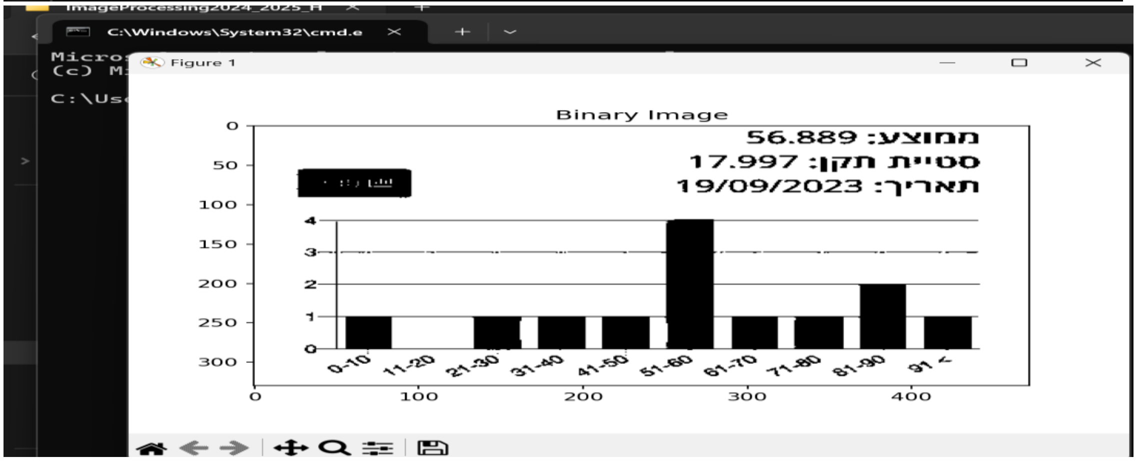


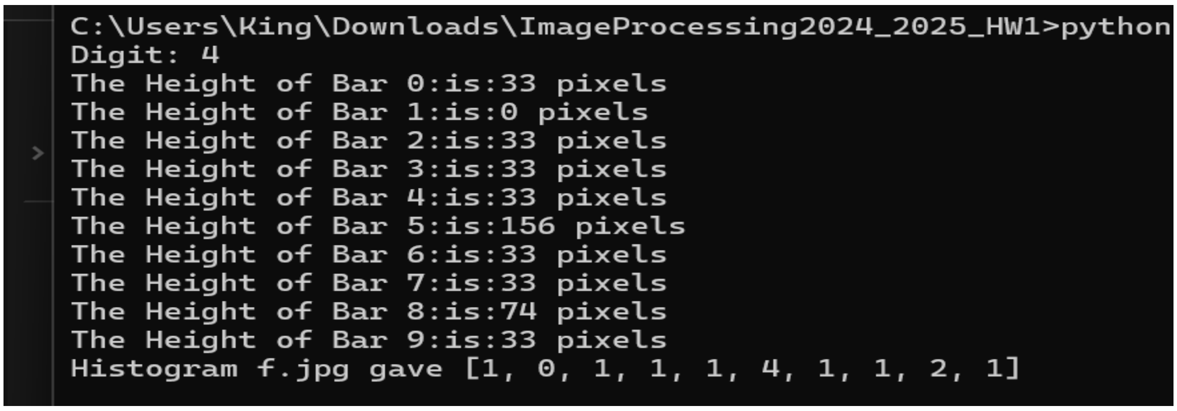




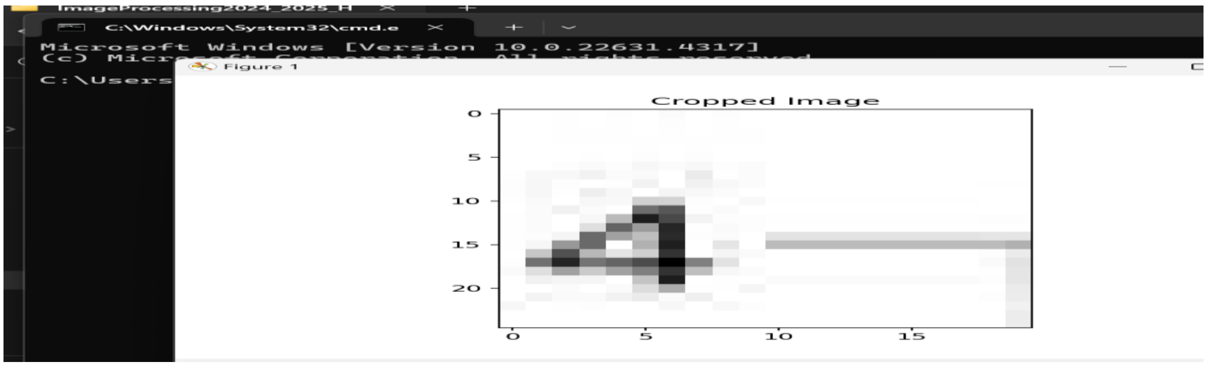
sixth image :



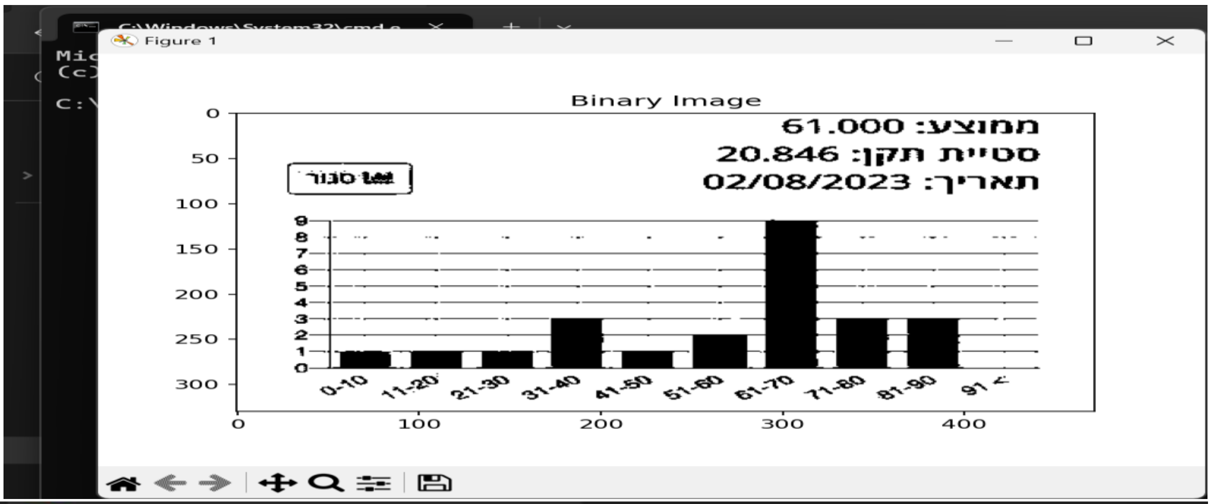


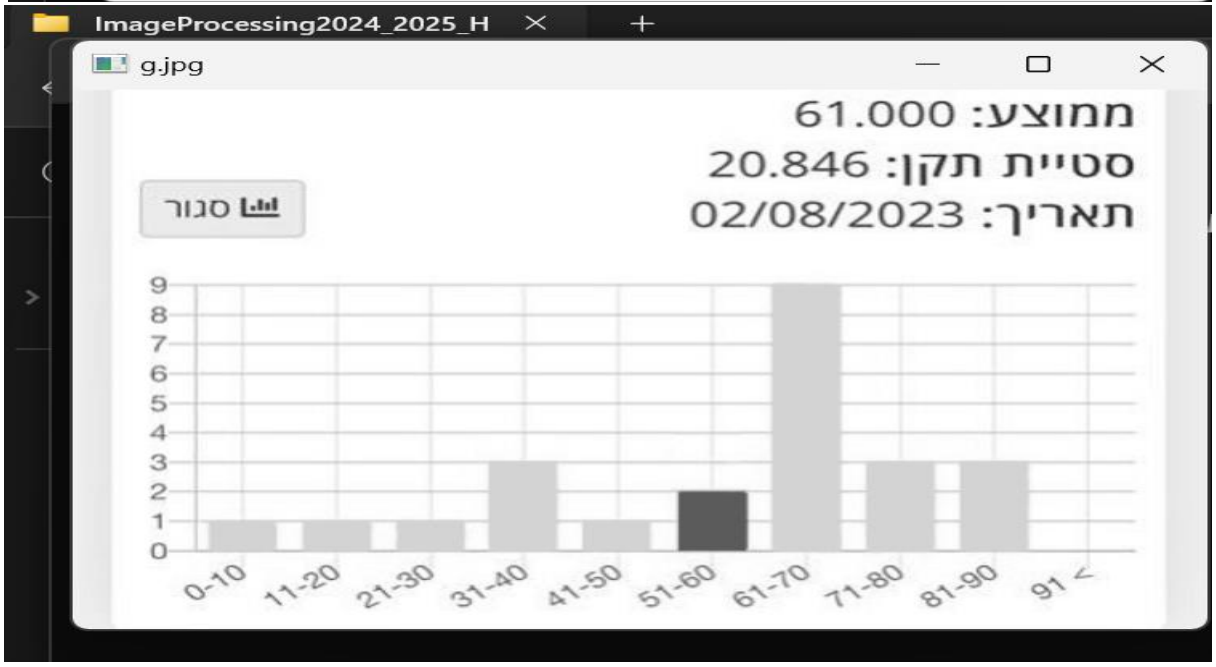


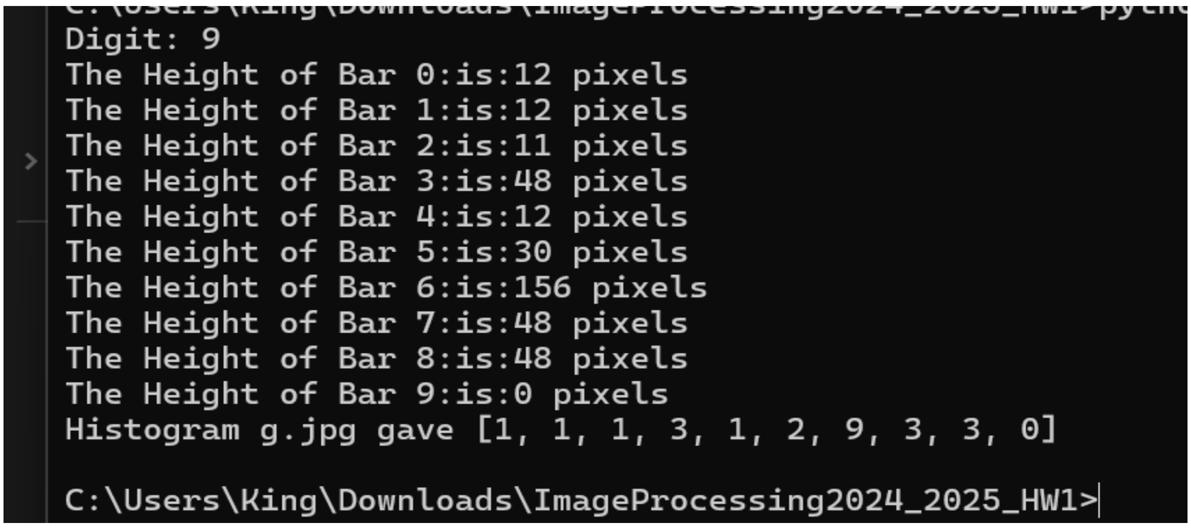
```
C:\Users\King\Downloads\ImageProcessing2024_2025_HW1>python
Digit:  4
The Height of Bar 0:is:33 pixels
The Height of Bar 1:is:0 pixels
The Height of Bar 2:is:33 pixels
The Height of Bar 3:is:33 pixels
The Height of Bar 4:is:33 pixels
The Height of Bar 5:is:156 pixels
The Height of Bar 6:is:33 pixels
The Height of Bar 7:is:33 pixels
The Height of Bar 8:is:74 pixels
The Height of Bar 9:is:33 pixels
Histogram f.jpg gave [1, 0, 1, 1, 1, 4, 1, 1, 2, 1]
```



seventh image :







```
Digit: 9
The Height of Bar 0:is:12 pixels
The Height of Bar 1:is:12 pixels
The Height of Bar 2:is:11 pixels
The Height of Bar 3:is:48 pixels
The Height of Bar 4:is:12 pixels
The Height of Bar 5:is:30 pixels
The Height of Bar 6:is:156 pixels
The Height of Bar 7:is:48 pixels
The Height of Bar 8:is:48 pixels
The Height of Bar 9:is:0 pixels
Histogram g.jpg gave [1, 1, 1, 3, 1, 2, 9, 3, 3, 0]

C:\Users\King\Downloads\ImageProcessing2024_2025_HW1>
```