HW1-ICA - Minh Nguyen #2069407

In [151…
```
import cv2
from matplotlib import pyplot as plt
```

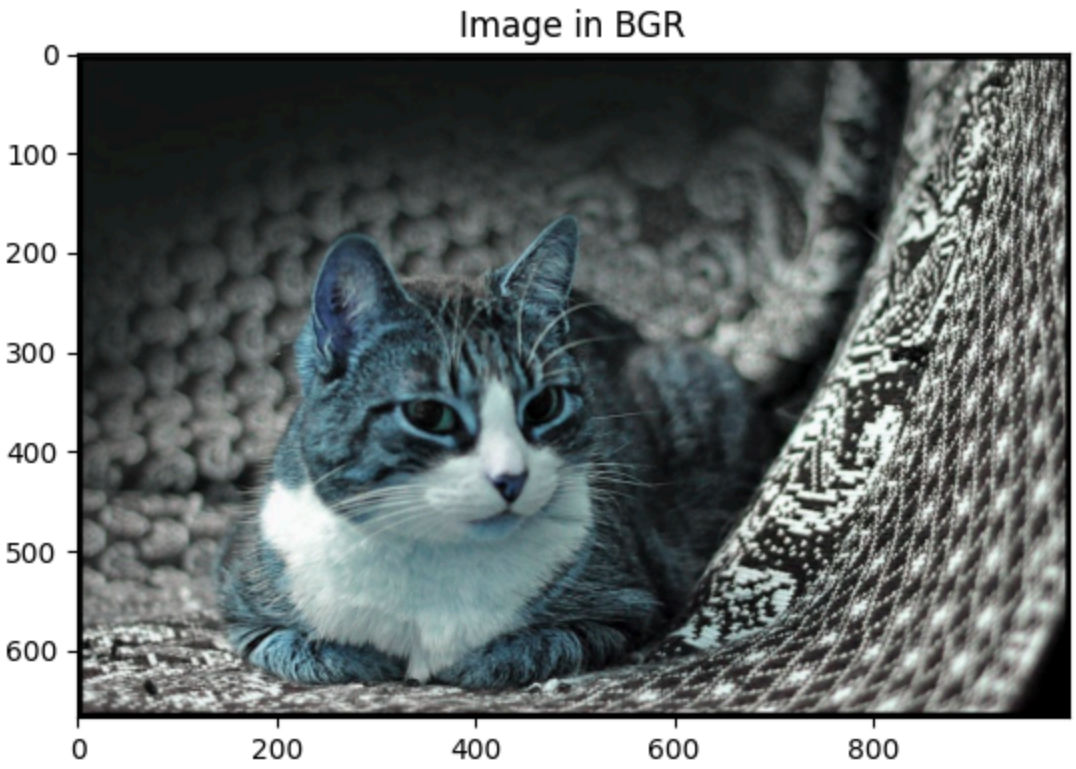
Task a:

- Load an image using OpenCV convert it from BGR to an RGB color format
- Display the image using matplotlib

In [152…
```
img = cv2.imread("cat_on_sofa.png")
```

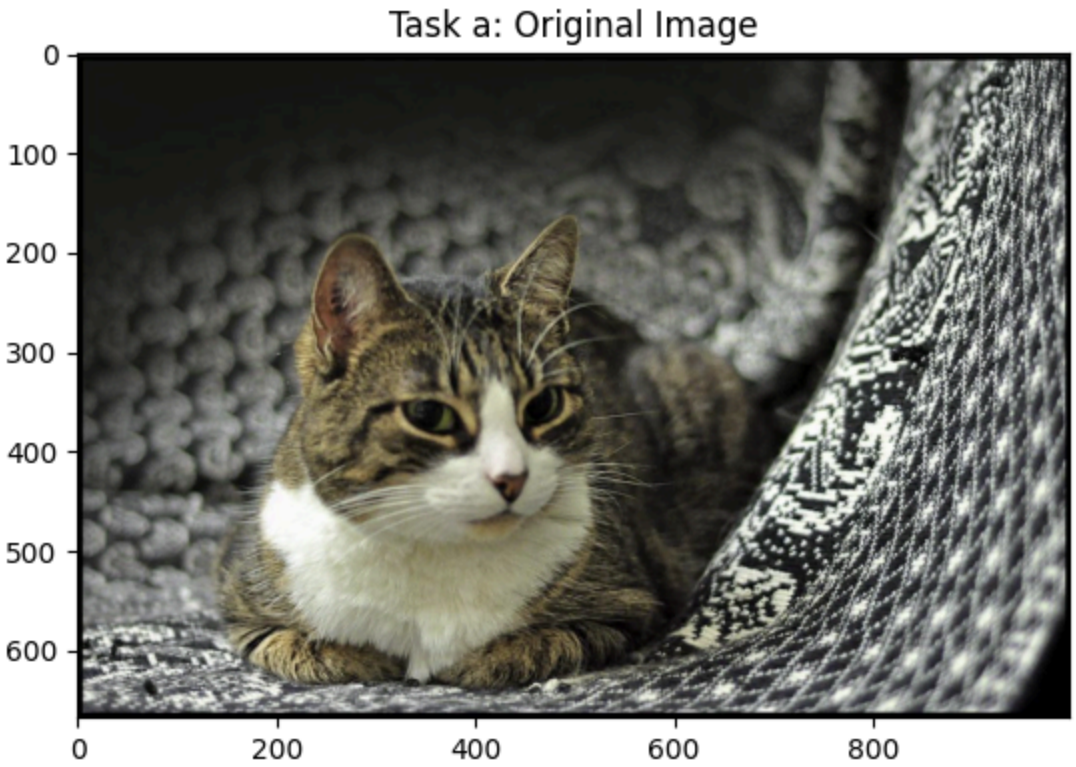
In [153…
```
# display the image with matplotlib
plt.imshow(img)
plt.title("Image in BGR")
```

Out[153… `Text(0.5, 1.0, 'Image in BGR')`



In [154…
```
# convert from BGR to RGB
img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
# display the image with matplotlib
plt.imshow(img_rgb)
plt.title("Task a: Original Image")
```

Out[154… `Text(0.5, 1.0, 'Task a: Original Image')`



Task b:

- Print (i) the number of pixels and (ii) the height, width, and number of channels of the

loaded image.

In [155…
```
print("Task b:")
print("- Number of pixels: ", str(img.size))
print("- Shape/Dimensions: " + str(img.shape))
print("- Image height: ", str(img.shape[0]))
print("- Image width: ", str(img.shape[1]))
print("- Number of channels: ", str(img.shape[2]))
```

```
Task b:
- Number of pixels:  1999992
- Shape/Dimensions: (668, 998, 3)
- Image height:  668
- Image width:  998
- Number of channels:  3
```

Task c:

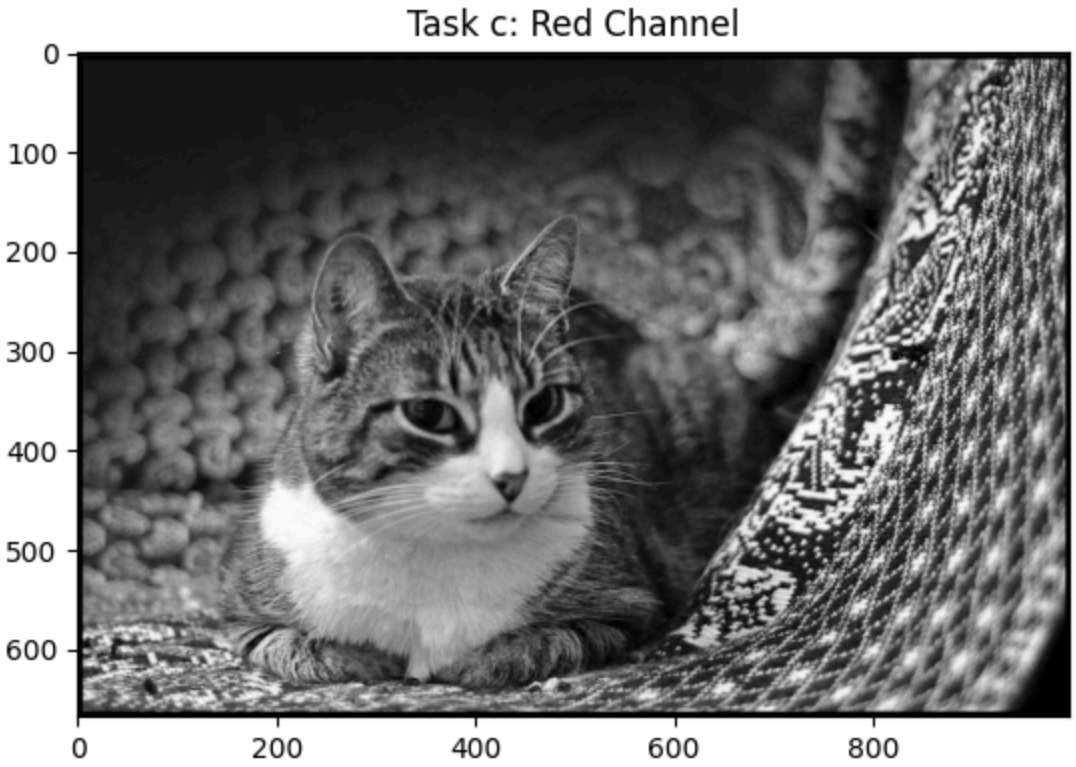
- Isolate the information of the Red, Green, and Blue channels of the image and plot them

in the Jupyter Notebook using Matplotlib

In [156…
```
# split the image into 3 channels
red, green, blue = cv2.split(img_rgb)
```

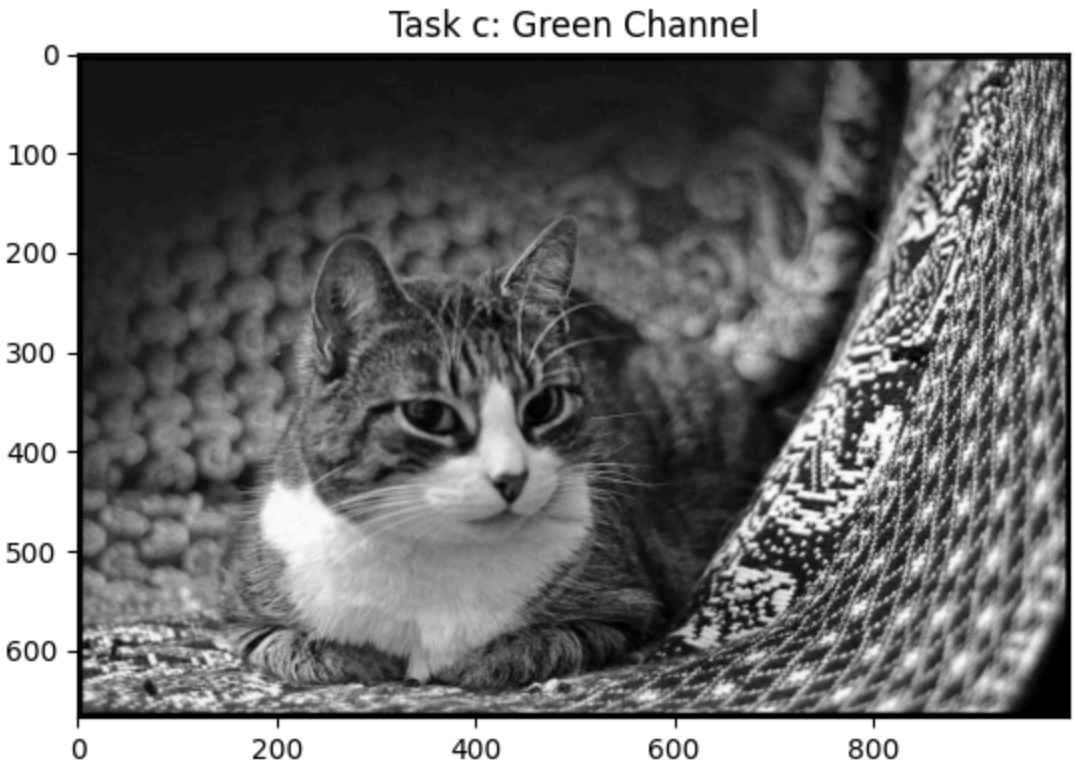
In [157…
```
plt.imshow(red, cmap='gray')
plt.title('Task c: Red Channel')
```

Out[157… Text(0.5, 1.0, 'Task c: Red Channel')



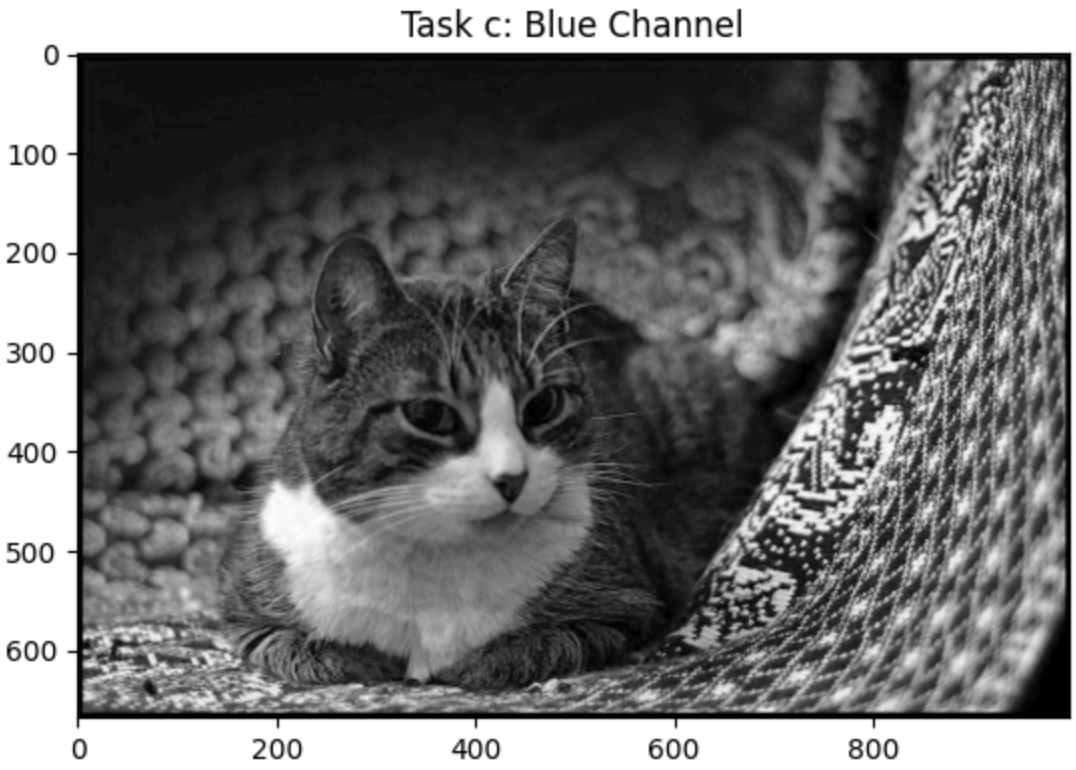
In [158…
```
plt.imshow(green, cmap='gray')
plt.title('Task c: Green Channel')
```

Out[158… Text(0.5, 1.0, 'Task c: Green Channel')



In [159…
```
plt.imshow(blue, cmap='gray')
plt.title('Task c: Blue Channel')
```

Out[159… `Text(0.5, 1.0, 'Task c: Blue Channel')`

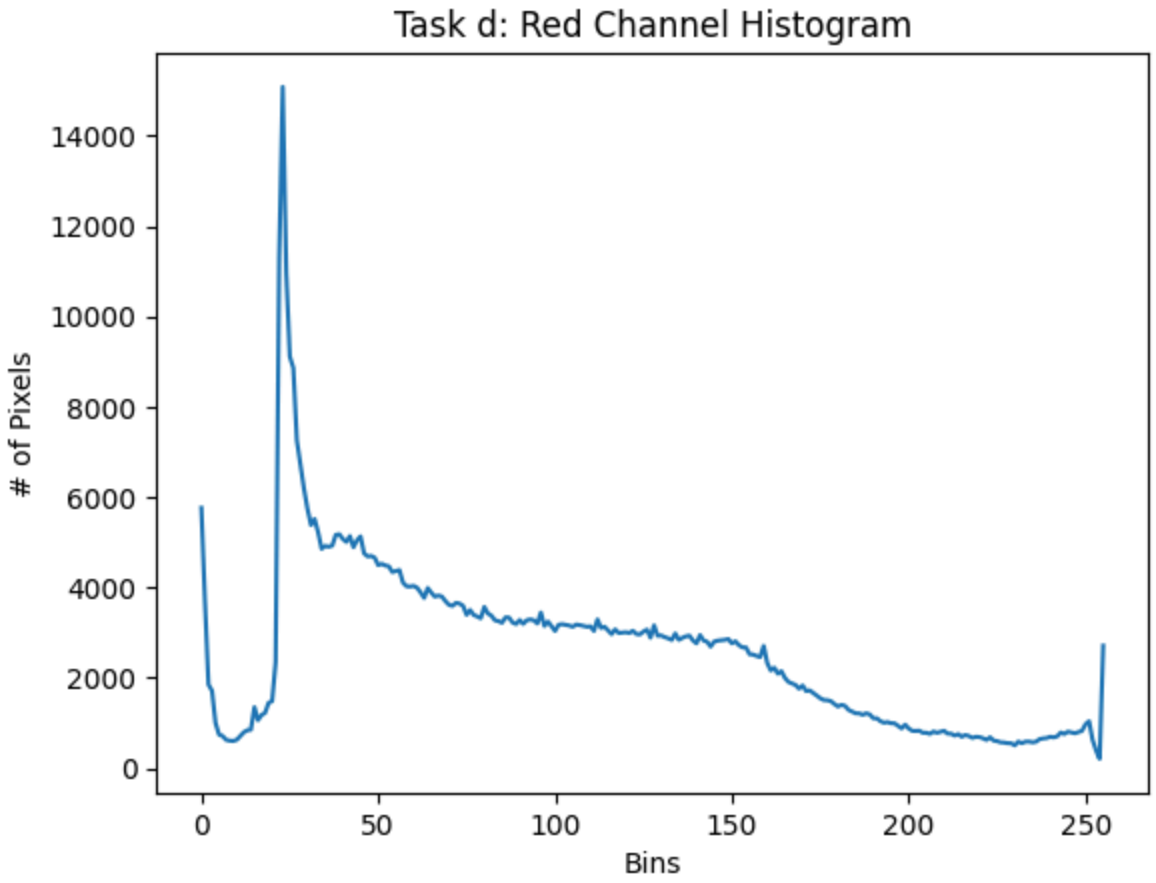


Task d:

- Calculate the unnormalized histograms of the Red, Green, and Blue channels of the image and plot them using Matplotlib

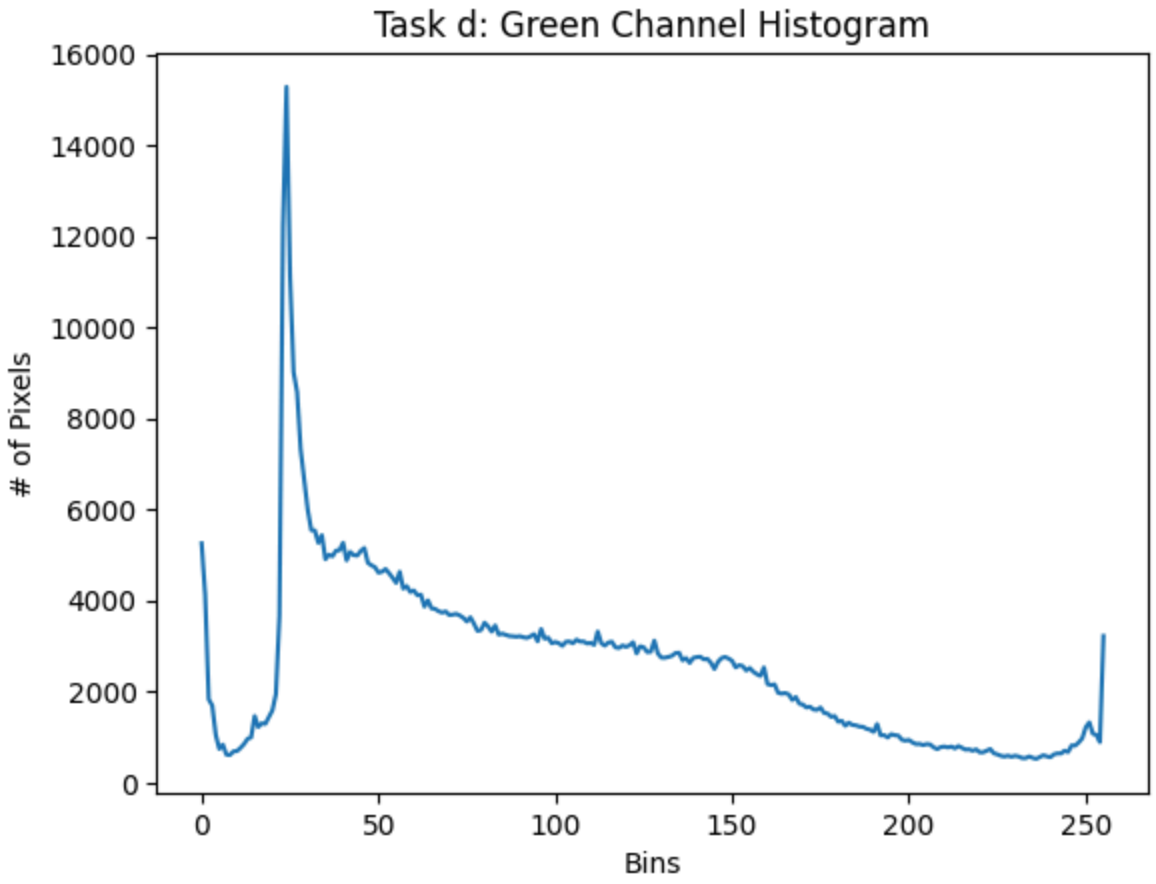
In [160…
```
# calculate the histogram of the red channel
red_hist = cv2.calcHist([red], [0], None, [256], [0, 256])
plt.plot(red_hist)
plt.xlabel('Bins')
plt.ylabel('# of Pixels')
plt.title('Task d: Red Channel Histogram')
```

Out[160…
```
Text(0.5, 1.0, 'Task d: Red Channel Histogram')
```



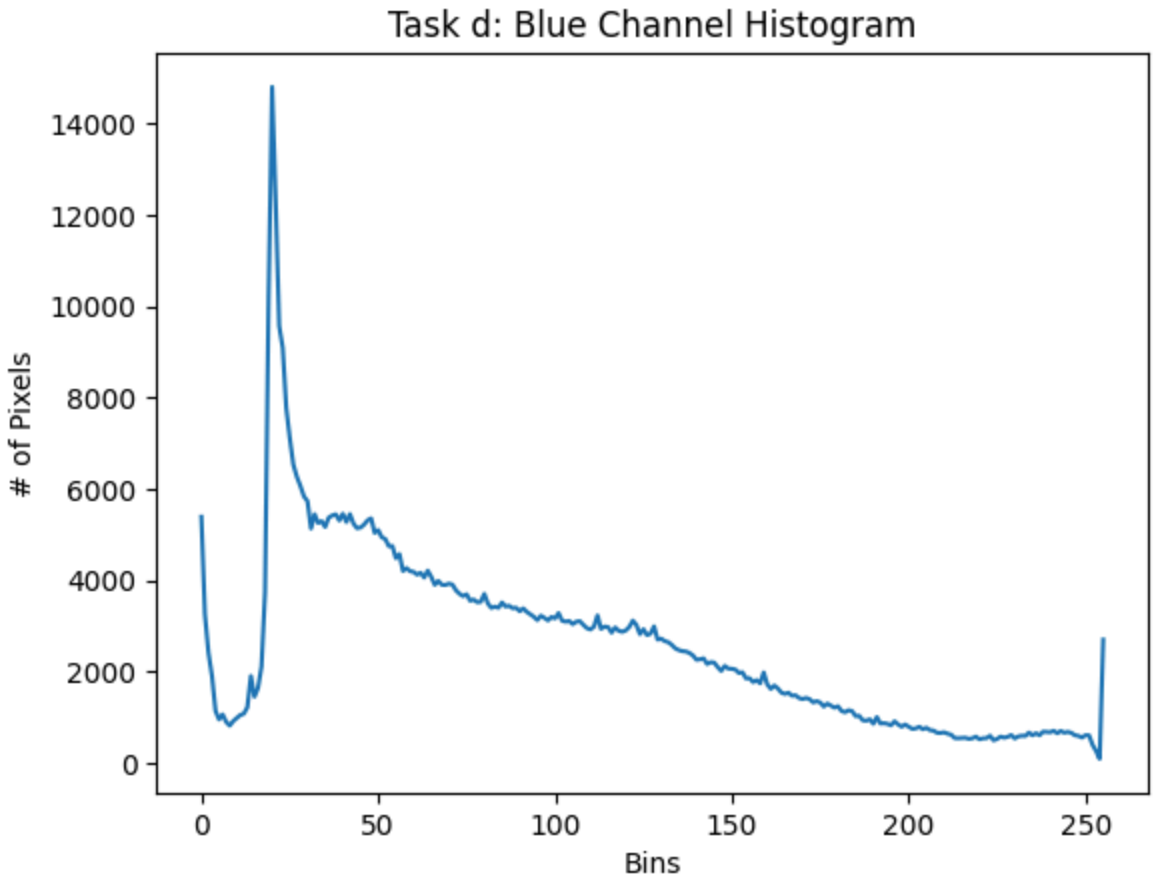
In [161…
```
# calculate the histogram of the green channel
green_hist = cv2.calcHist([green], [0], None, [256], [0, 256])
plt.plot(green_hist)
plt.xlabel('Bins')
plt.ylabel('# of Pixels')
plt.title('Task d: Green Channel Histogram')
```

Out[161…
```
Text(0.5, 1.0, 'Task d: Green Channel Histogram')
```



In [162…
```
# calculate the histogram of the blue channel
blue_hist = cv2.calcHist([blue], [0], None, [256], [0, 256])
plt.plot(blue_hist)
plt.xlabel('Bins')
plt.ylabel('# of Pixels')
plt.title('Task d: Blue Channel Histogram')
```

Out[162…
```
Text(0.5, 1.0, 'Task d: Blue Channel Histogram')
```

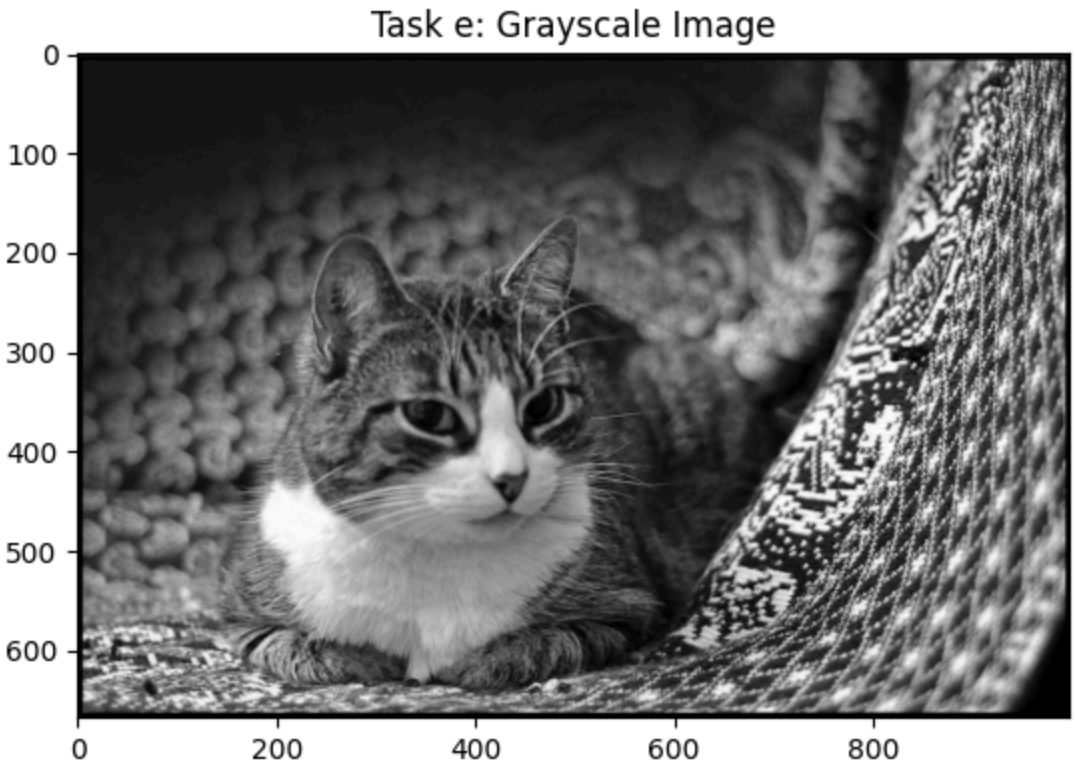


Task e:

- Convert the image to grayscale and plot it in the Jupyter Notebook using Matplotlib

In [163…
```
# convert the image to grayscale
img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
plt.imshow(img_gray, cmap='gray')
plt.title("Task e: Grayscale Image")
```

Out[163…
```
Text(0.5, 1.0, 'Task e: Grayscale Image')
```



Task f:

- Print (i) the number of pixels and (ii) the height, width, and number of channels of the

grayscale image

In [164…
```
print("Task f:")
print("- Number of pixels: ", str(img_gray.size))
print("- Shape/Dimensions: " + str(img_gray.shape))
print("- Image height: ", str(img_gray.shape[0]))
print("- Image width: ", str(img_gray.shape[1]))
```

```
Task f:
- Number of pixels:  666664
- Shape/Dimensions: (668, 998)
- Image height:  668
- Image width:  998
```

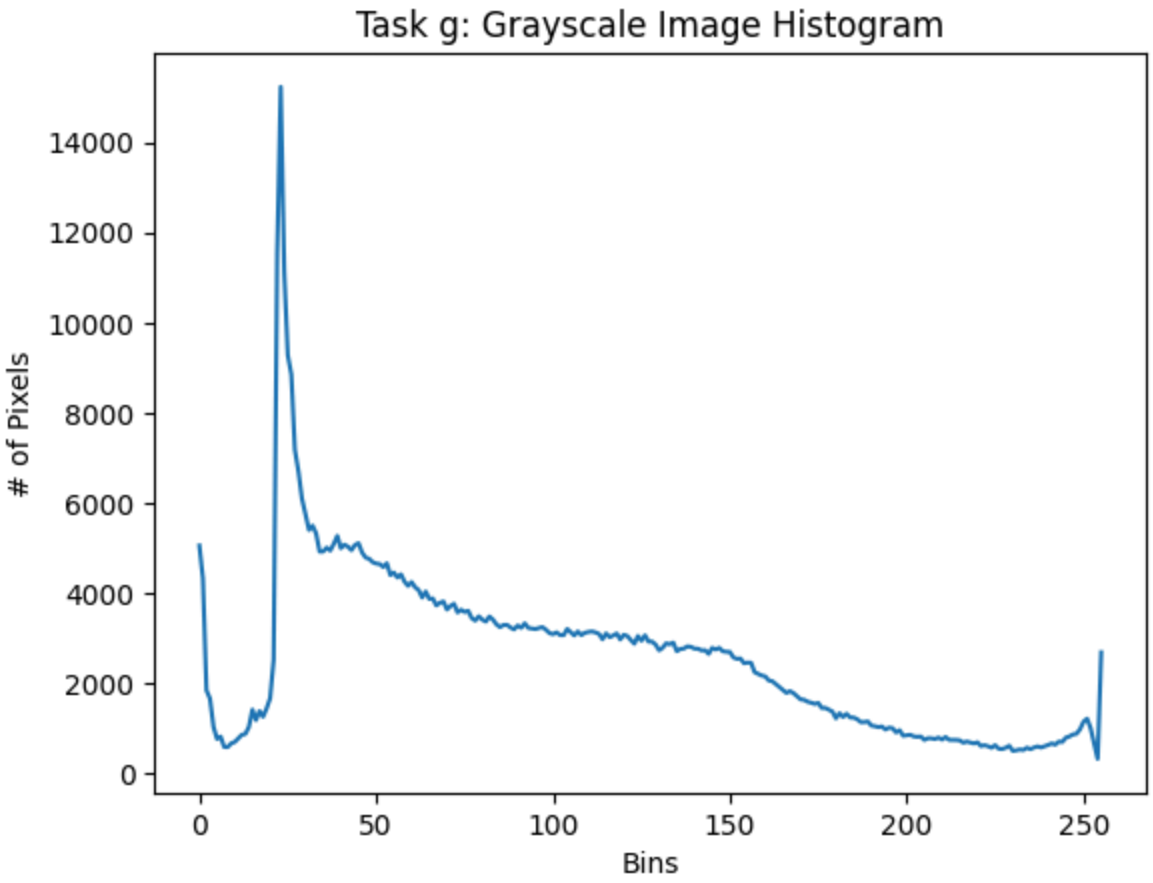
Task g:

- Calculate the unnormalized histogram of the grayscale image and plot it using Matplotlib

In [165…
```
gray_hist = cv2.calcHist([img_gray], [0], None, [256], [0, 256])
plt.plot(gray_hist)
plt.xlabel('Bins')
```

```
plt.ylabel('# of Pixels')
plt.title('Task g: Grayscale Image Histogram')
```

Out[165… `Text(0.5, 1.0, 'Task g: Grayscale Image Histogram')`



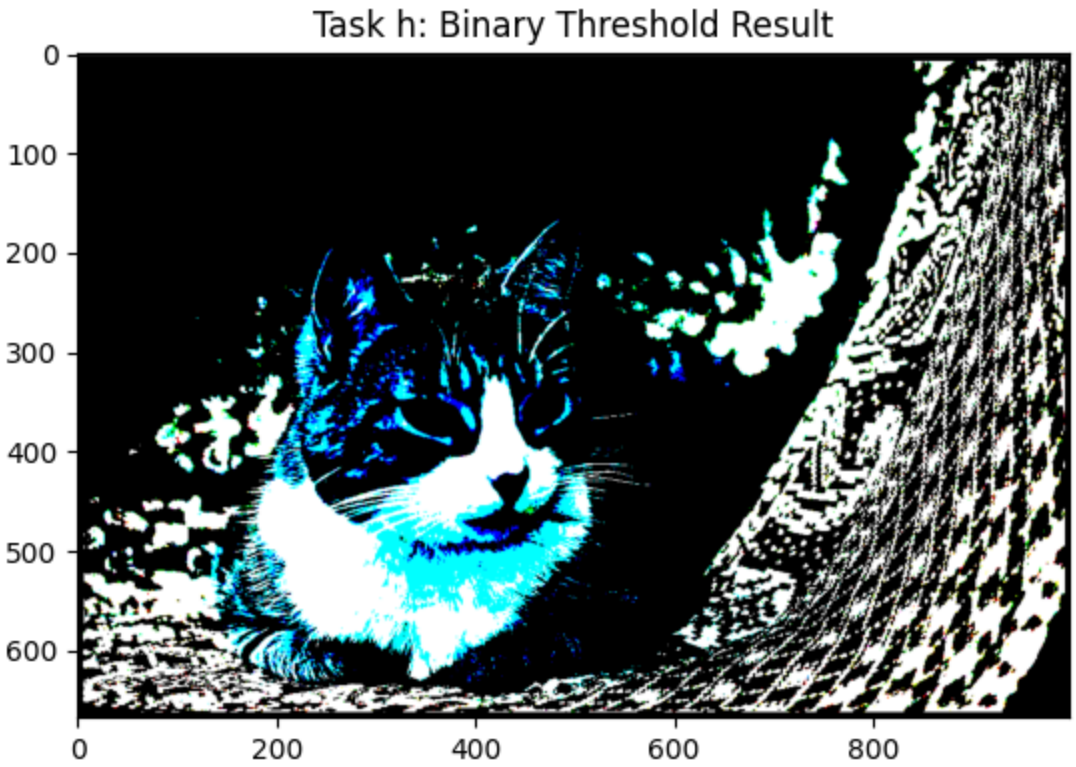
Task h:

- Use a binary threshold on your grayscale image and plot the result in the Jupyter

Notebook using Matplotlib

In [166…
```
# use a binary threshold with a threshold value of 128
r, threshold = cv2.threshold(img, 128, 255, cv2.THRESH_BINARY)
plt.imshow(threshold)
plt.title("Task h: Binary Threshold Result")
```

Out[166… `Text(0.5, 1.0, 'Task h: Binary Threshold Result')`



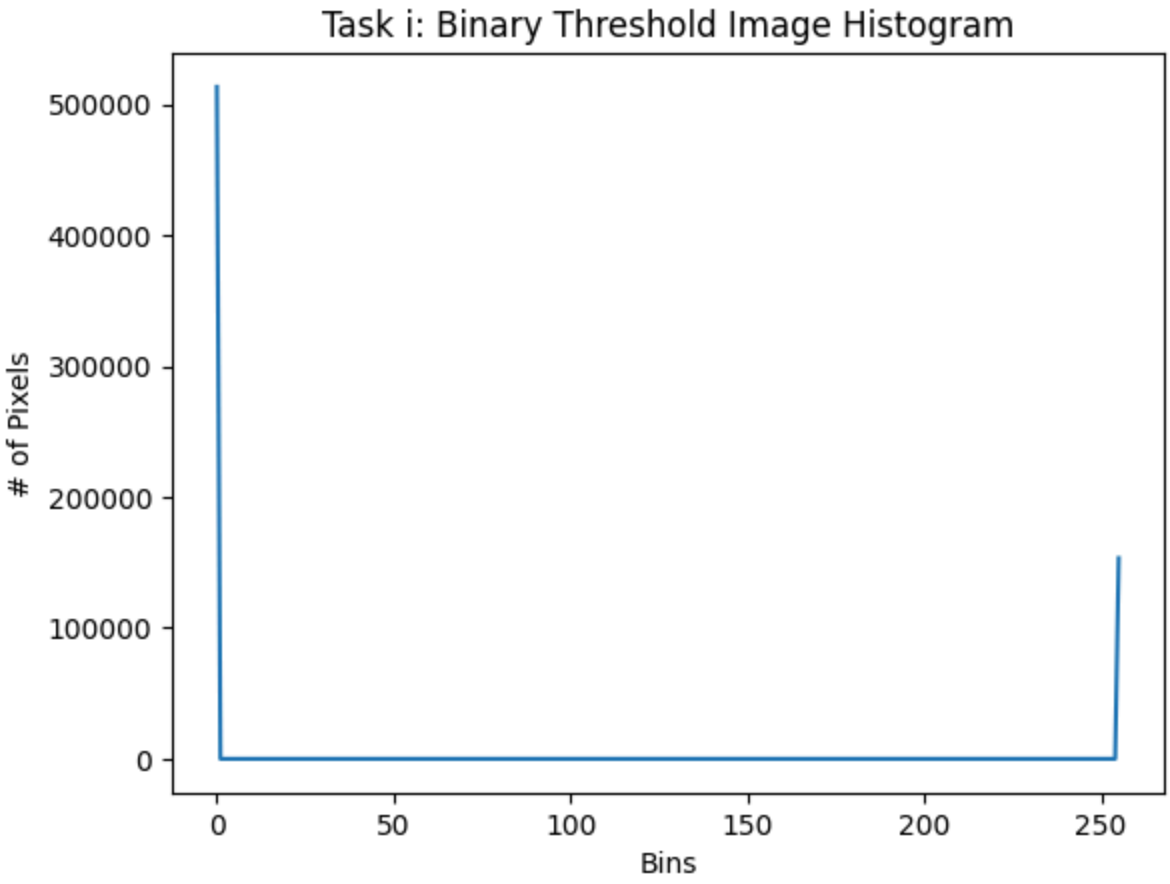
Task i:

- Calculate the unnormalized histogram of the thresholded image and plot it using

Matplotlib

In [167…
```
# calculate the histogram of the binary threshold image
threshold_hist = cv2.calcHist([threshold], [0], None, [256], [0, 256])
plt.plot(threshold_hist)
plt.xlabel('Bins')
plt.ylabel('# of Pixels')
plt.title('Task i: Binary Threshold Image Histogram')
```

Out[167…
```
Text(0.5, 1.0, 'Task i: Binary Threshold Image Histogram')
```



Task j: Commentary Section

i) What are the differences you observe in the histograms for the different color channels?

- All 3 channel histograms have a similar shape, but they do have some small differences in the distribution of intensity values.
- They are slightly different in the bins from 150 to 255, where the blue channel has more intensity values than the red and green channels.
- We can also observe the differences when splitting the color channels and plotting them separately. The blue channel are "darker" than the red and green channels.

ii) Converting an image to grayscale is the same operation as isolating a color channel? Yes/No and why?

- No, because the grayscale image is a single channel image, while isolating a color channel

extracts the values of a specific color (red, green or blue) from a multi-channel image.

iii) What are the differences you observe in the histograms for the grayscale image and the thresholded image?

- In the grayscale histogram, the intensity values are distributed across the entire range of the image.
- In contrast, in the thresholded histogram, the intensity values are only present at the two ends, 0 and 255, because the binary thresholding operation assigns the pixel values either 0 or 1.