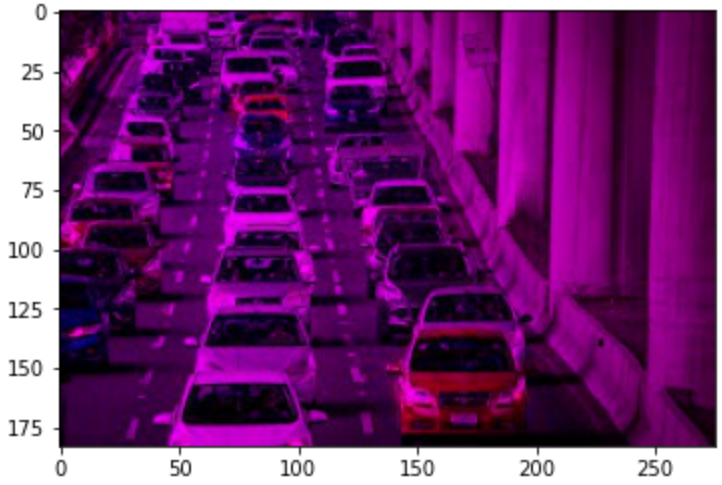
In [22]:
```
import cv2
import matplotlib.pyplot as plt
import numpy as np
```

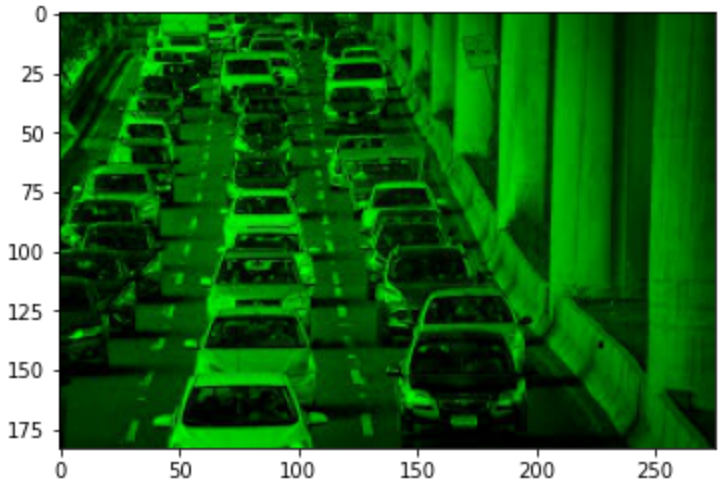
Problema 1 </h2>

In [19]:
```
def canal_imagen(imagen, color):
    img = cv2.imread(imagen) #lectura en formato BGR
    img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
    alto = img.shape[0]
    ancho =  img.shape[1]
    lienzo = np.zeros((alto,ancho,3)) #hace que la imagen de salida sea a color.
    for i in range(0, alto):
        for j in range(0, ancho):
            pixel = img[i,j]

            blue = pixel[2]
            green = pixel[1]
            red = pixel[0]

            if(color == 1):
                lienzo[i, j] = [blue, 0, 0]
            if(color == 2):
                lienzo[i, j] = [0, green, 0]
            if(color == 3):
                lienzo[i, j] = [0, 0, red]
            if(color == 10):
                lienzo[i, j] = [0, green, red]
            if(color == 20):
                lienzo[i, j] = [blue, green, 0]
            if(color == 30):
                lienzo[i, j] = [blue, 0, red]

    cv2.imwrite('monocromo.jpg', lienzo)
    monocromo = cv2.imread('monocromo.jpg')
    monocromo = cv2.cvtColor(monocromo, cv2.COLOR_BGR2RGB)
    return monocromo
```

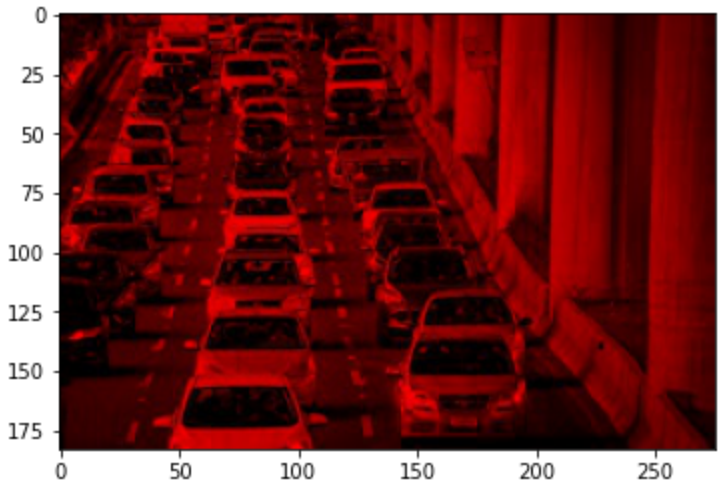
In [20]:
```
imagen_dir = 'image.jpg'
plt.imshow(canal_imagen(imagen_dir, 1))
plt.show()
```



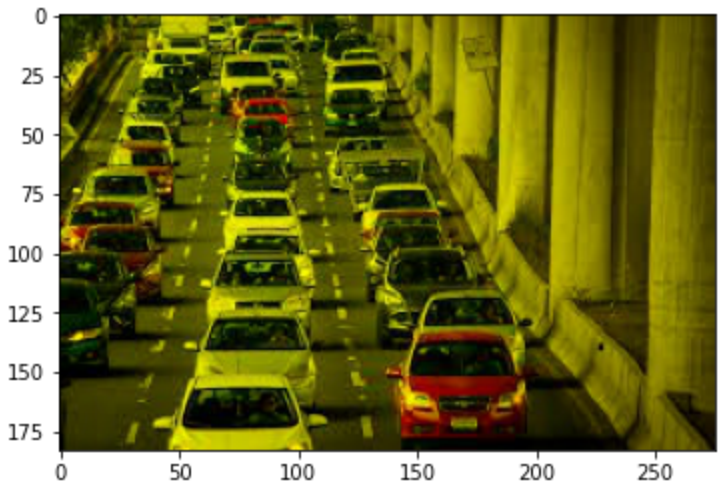
In [61]:
```
imagen_dir = 'image.jpg'
plt.imshow(canal_imagen(imagen_dir, 2))
plt.show()
```



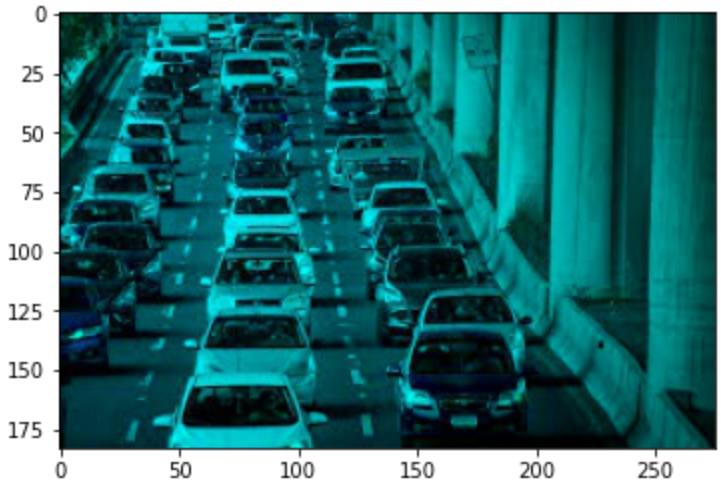
In [62]:
```
imagen_dir = 'image.jpg'
plt.imshow(canal_imagen(imagen_dir, 3))
plt.show()
```



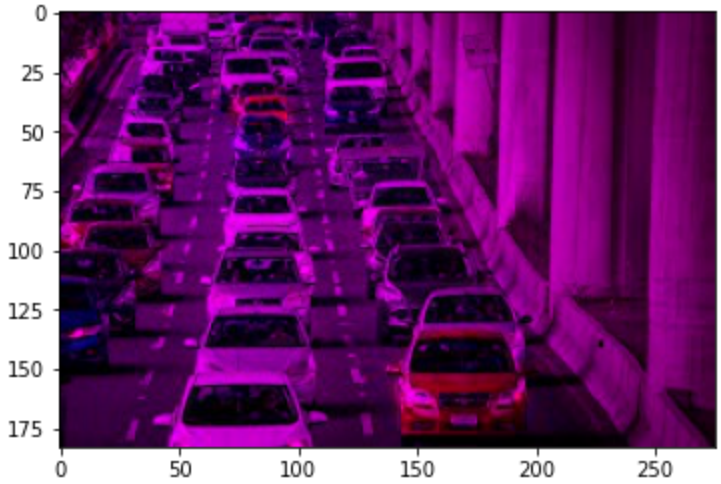
In [63]:
```
imagen_dir = 'image.jpg'
plt.imshow(canal_imagen(imagen_dir, 10))
plt.show()
```



In [64]:
```
imagen_dir = 'image.jpg'
plt.imshow(canal_imagen(imagen_dir, 20))
plt.show()
```



In [65]:
```
imagen_dir = 'image.jpg'
plt.imshow(canal_imagen(imagen_dir, 30))
plt.show()
```



Problema 2 </h2>

In [44]:
```
def agregar_colores(imagenR,imagenG, imagenB):
    imgR = cv2.imread(imagenR) #lectura en formato BGR
    imgR = cv2.cvtColor(imgR,cv2.COLOR_BGR2RGB)

    imgG = cv2.imread(imagenG) #lectura en formato BGR
    imgG = cv2.cvtColor(imgG,cv2.COLOR_BGR2RGB)

    imgB = cv2.imread(imagenB) #lectura en formato BGR
    imgB = cv2.cvtColor(imgB,cv2.COLOR_BGR2RGB)

    altoR = imgR.shape[0]
    anchoR =  imgR.shape[1]

    altoG = imgG.shape[0]
    anchoG =  imgG.shape[1]

    altoB = imgB.shape[0]
    anchoB =  imgB.shape[1]

    lienzo = np.zeros((altoR,anchoR,3)) #hace que la imagen de salida sea a color.
    for i in range(0, altoR):
```

```
        for j in range(0, anchoR):
            pixelR = imgR[i,j]
            pixelG = imgG[i,j]
            pixelB = imgB[i,j]

            blueR = pixelR[2]
            greenR = pixelR[1]
            redR = pixelR[0]

            blueG = pixelG[2]
            greenG = pixelG[1]
            redG = pixelG[0]

            blueB = pixelB[2]
            greenB = pixelB[1]
            redB = pixelB[0]

            lienzo[i, j] = [blueB, greenG, redR]


    cv2.imwrite('imagen_original.jpg', lienzo)
    imagen_original = cv2.imread('imagen_original.jpg')
    imagen_original = cv2.cvtColor(imagen_original, cv2.COLOR_BGR2RGB)
    return imagen_original
```

In [45]:
```
imagenB_dir = 'imagen1/imagen1_salida_gray_azul.jpg'
imagenR_dir = 'imagen1/imagen1_salida_gray_rojo.jpg'
imagenG_dir = 'imagen1/imagen1_salida_gray_verde.jpg'
plt.imshow(agregar_colores(imagenR_dir,imagenG_dir,imagenB_dir))
plt.show()
```

In [46]:
```
imagen2B_dir = 'imagen2/imagen2_salida_gray_azul.jpg'
imagen2R_dir = 'imagen2/imagen2_salida_gray_rojo.jpg'
imagen2G_dir = 'imagen2/imagen2_salida_gray_verde.jpg'
plt.imshow(agregar_colores(imagen2R_dir,imagen2G_dir,imagen2B_dir))
plt.show()
```

In [47]:
```
imagenB_dir = 'perro/perro_salida_gray_azul.jpg'
imagenR_dir = 'perro/perro_salida_gray_rojo.jpg'
imagenG_dir = 'perro/perro_salida_gray_verde.jpg'
plt.imshow(agregar_colores(imagenR_dir,imagenG_dir,imagenB_dir))
plt.show()
```

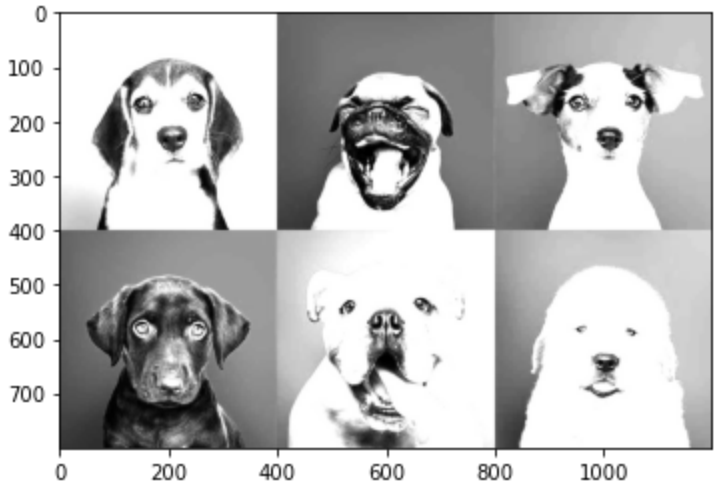
In [48]:
```
# imagenR_dir = 'imagen1/imagen1_salida_gray_rojo.jpg'
# imgR = cv2.imread(imagenB_dir)
# imgR
# plt.imshow(imgR) #DIBUJA EN rbg
# plt.show()
```

Problema 3 </h2>

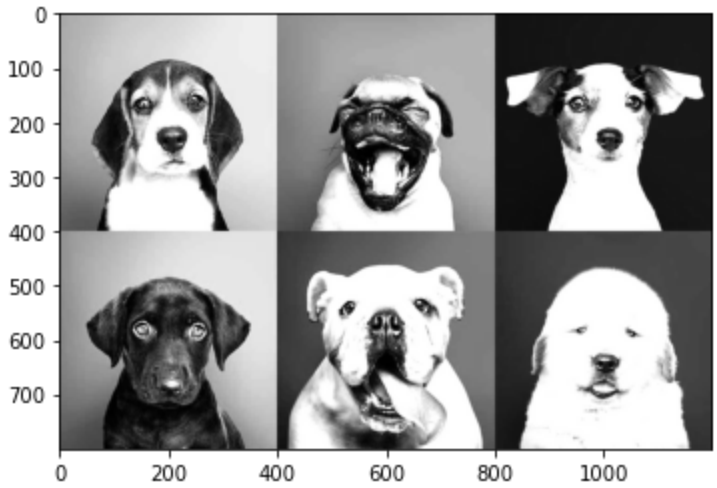
In [92]:
```
#IMAGEN ORIGINAL
imagen_a_gray = 'cachorros.jpg'
imgR = cv2.imread(imagen_a_gray)
imgR = cv2.cvtColor(imgR,cv2.COLOR_BGR2RGB)
imgR
plt.imshow(imgR) #DIBUJA EN rbg
plt.show()
```

In [54]:
```
def escala_gris_3D(imagen):
    img = cv2.imread(imagen) #lectura en formato BGR
    img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
    alto = img.shape[0]
    ancho =  img.shape[1]
    gris_red = np.zeros((alto,ancho,1))
    gris_green = np.zeros((alto,ancho,1))
    gris_blue = np.zeros((alto,ancho,1))
    for i in range(0, alto):
        for j in range(0, ancho):
            pixel = img[i,j]

            blue = pixel[2]
            green = pixel[1]
            red = pixel[0]

            gris_red[i,j] = int(0.299*blue + 0.587*green + red)
            gris_green[i,j] = int(0.299*blue + green + 0.114*red)
            gris_blue[i,j] = int(blue + 0.587*green + 0.114*red)

    cv2.imwrite('imagen_gray_red.jpg', gris_red)
    cv2.imwrite('imagen_gray_green.jpg', gris_green)
    cv2.imwrite('imagen_gray_blue.jpg', gris_blue)

    gris_red = cv2.imread('imagen_gray_red.jpg')
    gris_red = cv2.cvtColor(gris_red, cv2.COLOR_BGR2RGB)

    gris_green = cv2.imread('imagen_gray_green.jpg')
    gris_green = cv2.cvtColor(gris_green, cv2.COLOR_BGR2RGB)

    gris_blue = cv2.imread('imagen_gray_blue.jpg')
    gris_blue = cv2.cvtColor(gris_blue, cv2.COLOR_BGR2RGB)

    return gris_red, gris_green, gris_blue
```

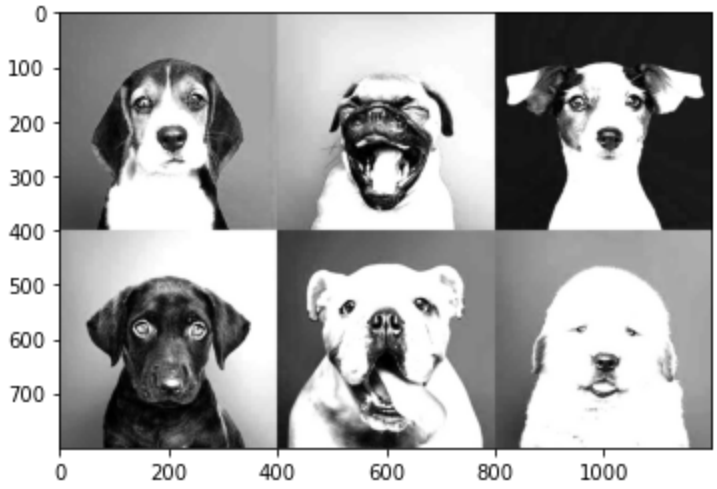
In [58]:
```
imagen_a_gray = 'cachorros.jpg'
plt.imshow(escala_gris_3D(imagen_a_gray)[0])
plt.show()
```



In [59]:
```
imagen_a_gray = 'cachorros.jpg'
plt.imshow(escala_gris_3D(imagen_a_gray)[1])
plt.show()
```



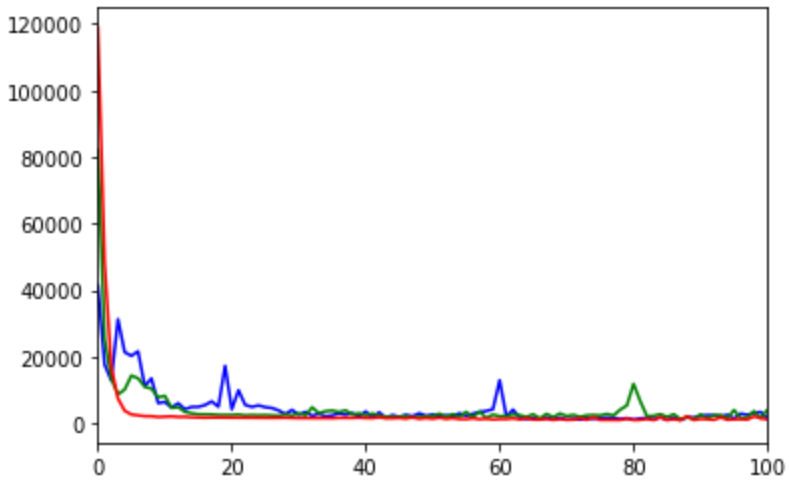
In [60]:
```
imagen_a_gray = 'cachorros.jpg'
plt.imshow(escala_gris_3D(imagen_a_gray)[2])
plt.show()
```



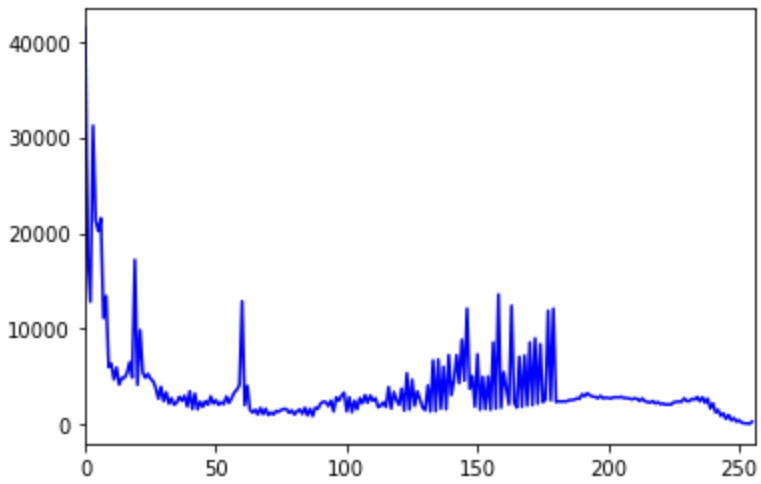
Problema 4 </h2>

In [71]:
```
image = cv2.imread('cachorros.jpg')
```

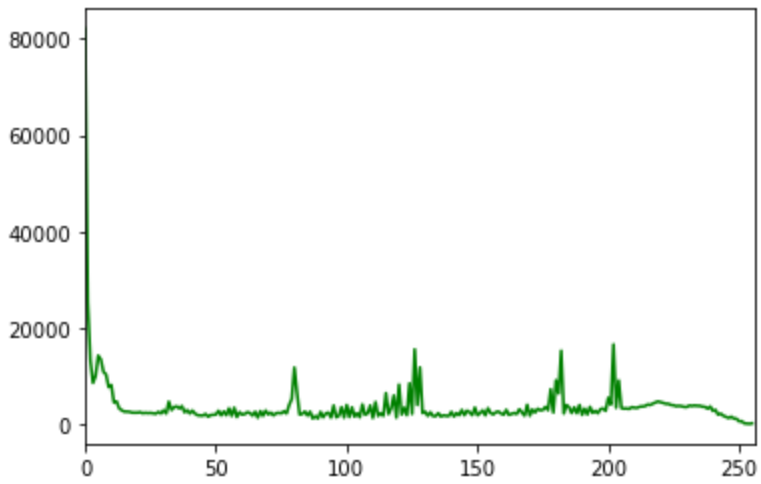
```
for i, col in enumerate(['b', 'g', 'r']):
    hist = cv2.calcHist([image], [i], None, [256], [0, 256])
    plt.plot(hist, color = col)
    plt.xlim([0, 256])

plt.show()
```



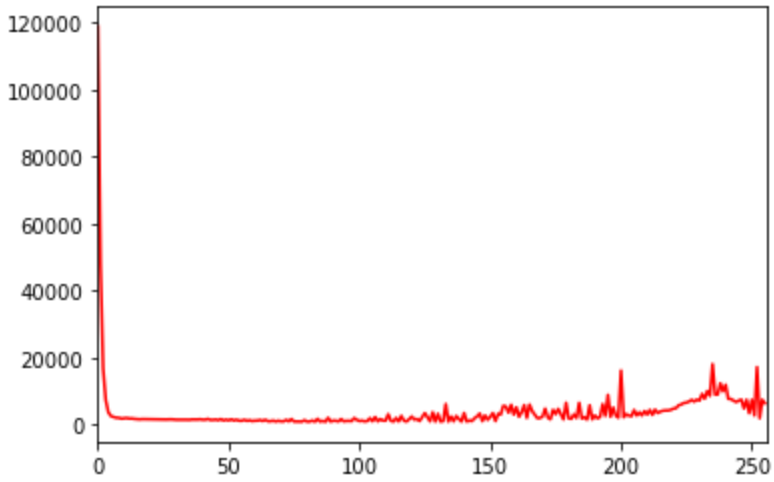
In [75]:
```
hist = cv2.calcHist([image], [0], None, [256], [0, 256])
plt.plot(hist, color = 'b')
plt.xlim([0, 256])
plt.show()
```



In [76]:
```
hist = cv2.calcHist([image], [1], None, [256], [0, 256])
plt.plot(hist, color = 'g')
plt.xlim([0, 256])
plt.show()
```



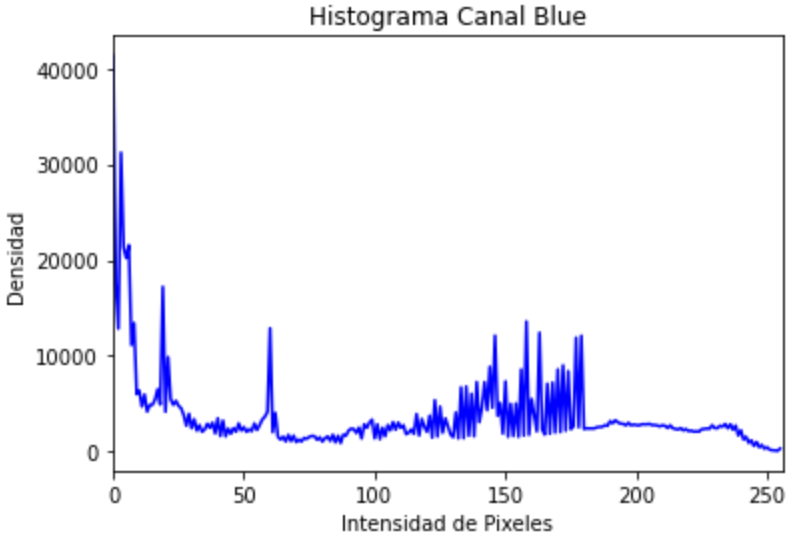
In [77]:
```
hist = cv2.calcHist([image], [2], None, [256], [0, 256])
plt.plot(hist, color = 'r')
plt.xlim([0, 256])
plt.show()
```

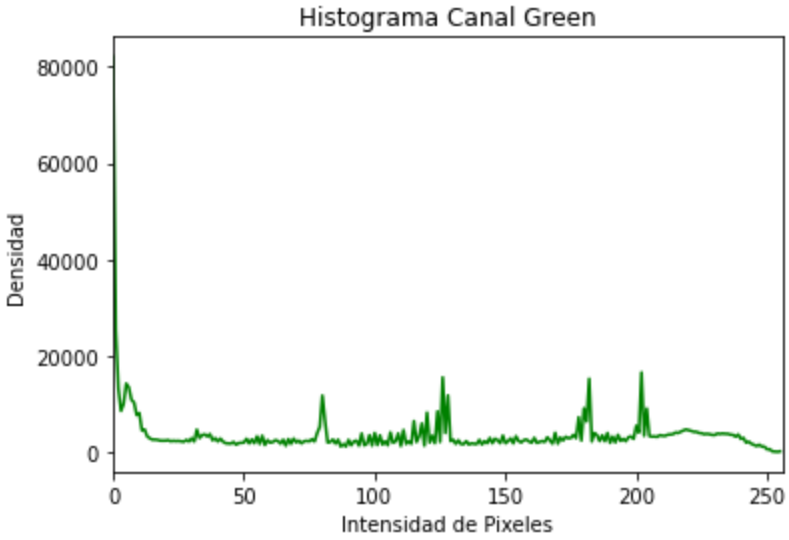


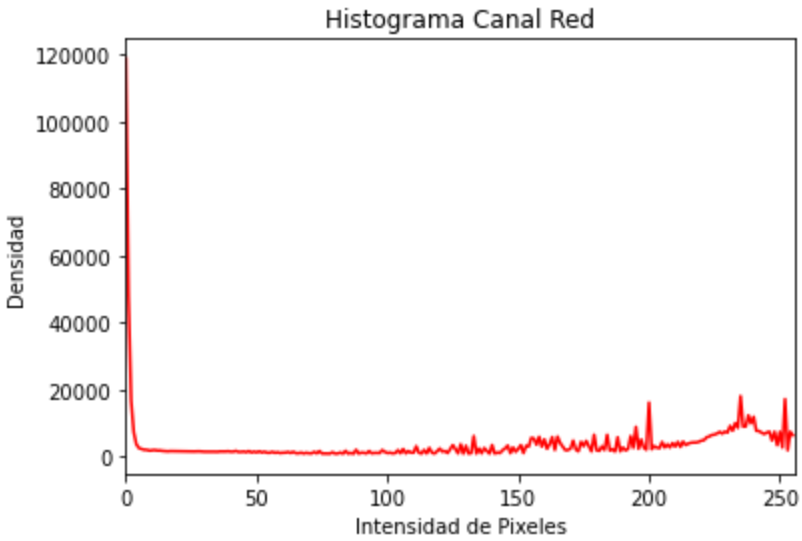
In [85]:
```
def histograma_x_canal(imagen):
    img = cv2.imread(imagen) #lectura en formato BGR
    img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
    alto = img.shape[0]
    ancho =  img.shape[1]
    gris_ponderado = np.zeros((alto,ancho,1))
    for i in range(0, alto):
        for j in range(0, ancho):
            pixel = img[i,j]

            blue = pixel[2]
            green = pixel[1]
            red = pixel[0]

            gris_ponderado[i,j] = int(0.299*blue + 0.587*green + 0.114*red)
    cv2.imwrite('gris_ponderado.jpg', gris_ponderado)
    gris_ponderado = cv2.imread('gris_ponderado.jpg')
    gris_ponderado = cv2.cvtColor(gris_ponderado, cv2.COLOR_BGR2RGB)
#     plt.imshow(gris_ponderado)
#     plt.show()
    hist_b = cv2.calcHist([image], [0], None, [256], [0, 256])
    plt.plot(hist_b, color = 'b')
    plt.xlim([0, 256])
    plt.xlabel("Intensidad de Pixeles")
    plt.ylabel("Densidad")
    plt.title("Histograma Canal Blue")
    plt.show()

    hist_g = cv2.calcHist([image], [1], None, [256], [0, 256])
    plt.plot(hist_g, color = 'g')
    plt.xlim([0, 256])
    plt.xlabel("Intensidad de Pixeles")
    plt.ylabel("Densidad")
    plt.title("Histograma Canal Green")
    plt.show()

    hist_r = cv2.calcHist([image], [2], None, [256], [0, 256])
    plt.plot(hist_r, color = 'r')
    plt.xlim([0, 256])
    plt.xlabel("Intensidad de Pixeles")
    plt.ylabel("Densidad")
    plt.title("Histograma Canal Red")
    plt.show()
```

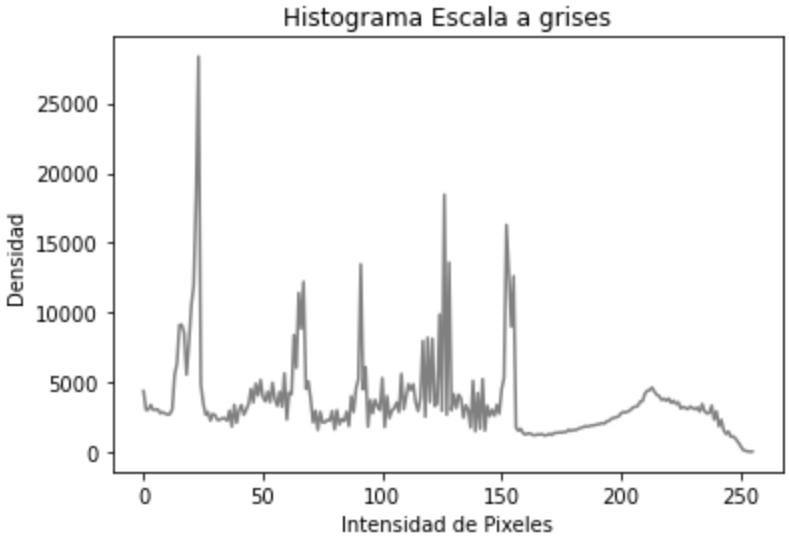
```
    hist_gray = cv2.calcHist([gris_ponderado], [0], None, [256], [0, 256])
    plt.plot(hist_gray, color="gray")
    plt.xlabel("Intensidad de Pixeles")
    plt.ylabel("Densidad")
    plt.title("Histograma Escala a grises")
    plt.show()
    return gris_ponderado
```

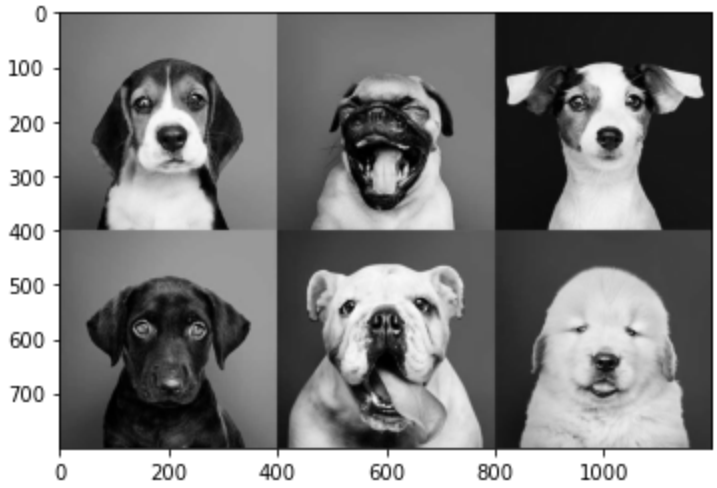
In [86]:
```
imagen = 'cachorros.jpg'
plt.imshow(histograma_x_canal(imagen))
plt.show()
```











Problema 5 </h2>

Escala a grises ponderado

La conversión de una imágen a escala a grises es la equivalencia a la luminancia de una imágen, por esto se realiza un cálculo del equivalente blanco y negro de una imágen con una media ponderada.

La ecuación de la luminancia nos muestra los factores de ponderación para cada canal de color y esto nos indica la sensibilidad del ojo humano a las frecuencias del espectro cercanas al rojo, ver y azul.

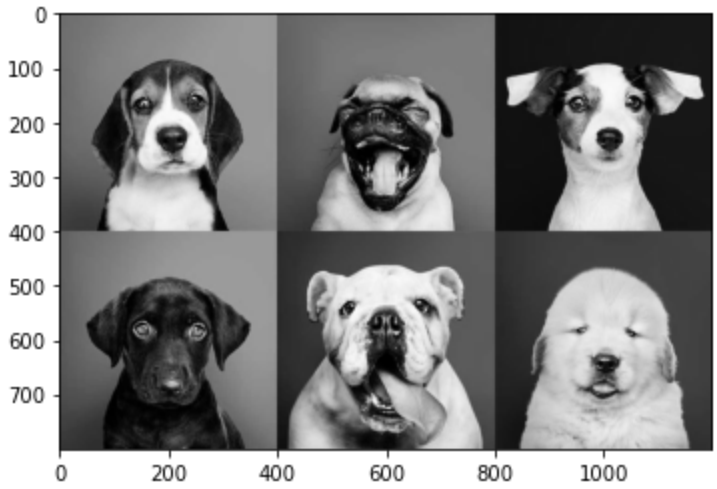
La ecuacion esta dada de la siguiente forma:

$$Y = R(0.3) + G(0.59) + B(0.11)$$

In [89]:
```
def gris_ponderado(imagen):
    img = cv2.imread(imagen) #lectura en formato BGR
    img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
    alto = img.shape[0]
    ancho =  img.shape[1]
    gris_ponderado = np.zeros((alto,ancho,1))
    for i in range(0, alto):
        for j in range(0, ancho):
```

```
            pixel = img[i,j]

            blue = pixel[2]
            green = pixel[1]
            red = pixel[0]

            gris_ponderado[i,j] = int(0.299*blue + 0.587*green + 0.114*red)
    cv2.imwrite('gris_ponderado.jpg', gris_ponderado)
    gris_ponderado = cv2.imread('gris_ponderado.jpg')
    gris_ponderado = cv2.cvtColor(gris_ponderado, cv2.COLOR_BGR2RGB)
    return gris_ponderado
```

In [90]:
```
imagen = 'cachorros.jpg'
plt.imshow(gris_ponderado(imagen))
plt.show()
```



Problema 6 </h2>

Espacio de Color HSV

El espacio de color HSV es una transformación no lineal del modelo RGB, que es una representación en tres dimensiones pero a diferencia del RGB, el HSV esta en coordenadas cilíndricas de manera que cada color viene definido por las siguientes características:

- **Tinte o Matiz:** Es el ángulo que representa el matiz, normalmente definido entre 0 y 360 grados
- **Saturación:** Es el nivel de saturación del color, dado entre 0 y 1, 0 representa sin saturación (blanco) y 1 sería el matiz en toda su intesidad.
- **Brillo** Es el nivel del brillo entre 0 y 1. 0 es negro y 1 es blanco.