In [12]:
```
import cv2
import numpy as np
```

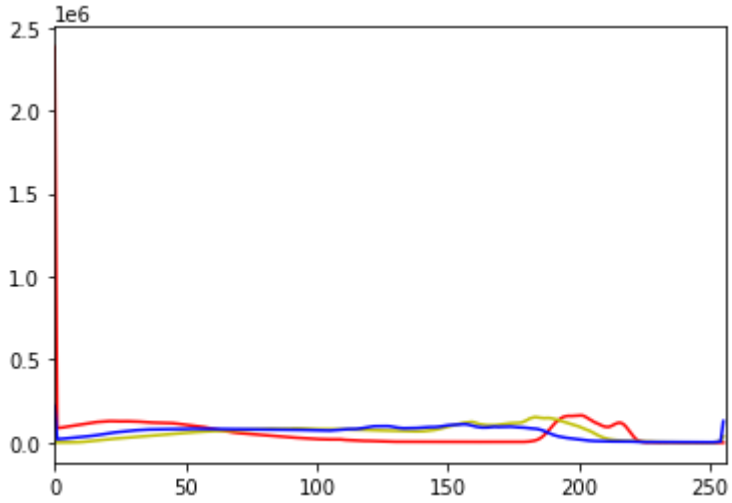
In [13]:
```
img = cv2.imread("mygirl with school.jpg")#读入一份图像
gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)#转化成灰度图
T,thresh = cv2.threshold(gray, 127, 255, cv2.THRESH_BINARY)
#由灰度图实现二值化的图片,T为阈值，cv2.THRESH_BINARY即为设置方法
cv2.imwrite("school.jpg", thresh)
```

Out[13]: True

In [15]:
```
import cv2
num_down = 2  # 缩减像素采样的数目
num_bilateral = 7 # 定义双边滤波的数目
img_rgb = cv2.imread("mygirl with school.jpg")
# 用高斯金字塔降低取样
img_color = img_rgb
for _ in range(num_down):
    img_color = cv2.pyrDown(img_color)
# 重复使用小的双边滤波代替一个大的滤波
for _ in range(num_bilateral):
    img_color = cv2.bilateralFilter(img_color, d=9,sigmaColor=9,sigmaSpace=7)
# 升采样图片到原始大小
for _ in range(num_down):
    img_color = cv2.pyrUp(img_color)
# 转换为灰度并使其产生中等的模糊
img_gray = cv2.cvtColor(img_rgb, cv2.COLOR_RGB2GRAY)
img_blur = cv2.medianBlur(img_gray, 7)
# 检测到边缘并且增强其效果
img_edge = cv2.adaptiveThreshold(img_blur, 255,cv2.ADAPTIVE_THRESH_MEAN_C,cv2.THRESH_
# 转换回彩色图像
img_edge = cv2.cvtColor(img_edge, cv2.COLOR_GRAY2RGB)
img_cartoon = cv2.bitwise_and(img_color, img_edge)
# 显示图片
cv2.imwrite("cartoon.jpg", img_cartoon)
```

Out[15]: True

In [16]:
```
import matplotlib.pyplot as plt
plt.figure()
channel = ["r","y","b"] #线的颜色
for i in range(3):#循环画出三种图
    fig = cv2.calcHist([img], [i], None, [256], [0,256])
    plt.plot(fig, color = channel[i])
    plt.xlim([0,256])
```



In [ ]: