Elaboración de un notebook básico de python donde se carguen los módulos de numpy y scikit image. Cargar una imagen de su disco duro.

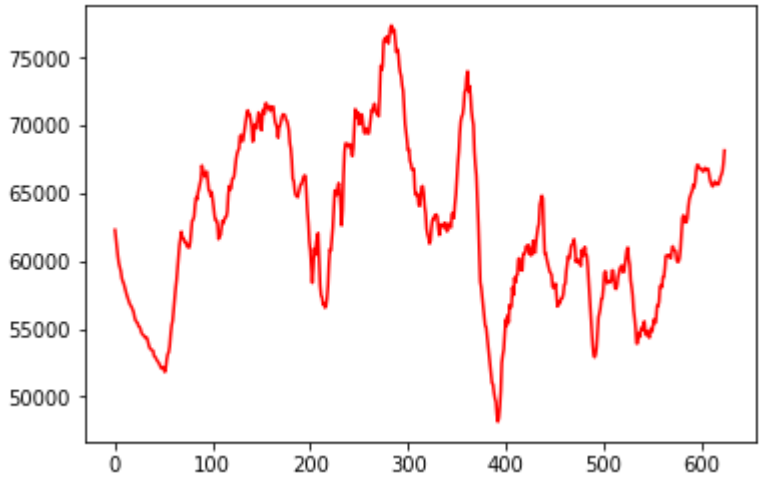
In [1]:
```
#importar librerías
import matplotlib.pyplot as plt
import numpy as np
from skimage import io
from skimage import data

#Cargar imagen
imtest1=data.imread("einstein1.jpg")
#Mostrar forma de imagen
print("Pixeles en X e Y: ", imtest1.shape)
```

```
Pixeles en X e Y:  (625, 640)
```

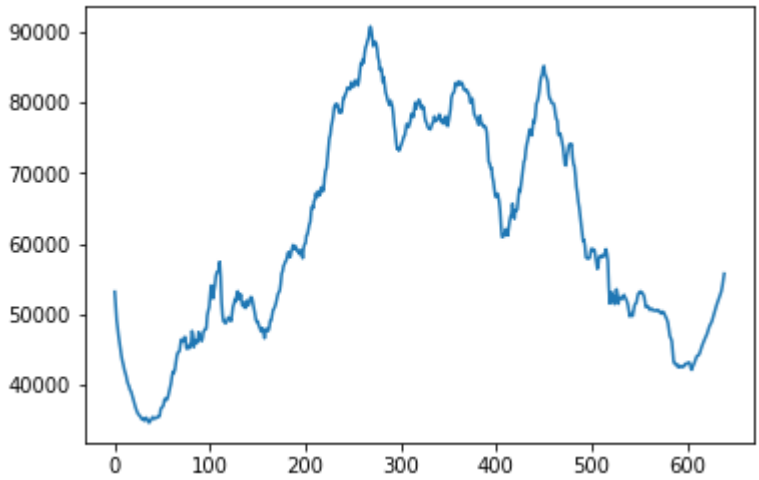
Obtener la distribución y su representación de la suma de las componentes de la imagen de forma vertical y horizontal. Posteriormente, se deberá representar dichos vectores haciendo uso de una librería de visualización como matplotlib.

In [2]:
```
#Suma de filas
sumfil= np.sum(imtest1, axis=1)

#Suma de columnas
sumcol= np.sum(imtest1, axis=0)



#Gráfica sumas
print("Gráfica de suma de filas: ")
#Gráfica( X= Número de pixeles en el eje x,Y=sumfil)
plt.plot(range(625), sumfil,'r')
plt.show()

print("Gráfica de suma de columnas: ")
#Gráfica( X= Número de pixeles en el eje Y,Y=sumcol)
plt.plot(range(640), sumcol)
plt.show()

#Máximo valor de la suma de filas y columnas
YMax=np.max(sumfil)
XMax=np.max(sumcol)
YMin=np.min(sumfil)
XMin=np.min(sumcol)
print("El valor máximo de las columnas y filas respectivamente es: ", XMax ,"
 y ",YMax)
print("El valor mínimo de las columnas y filas respectivamente es: ", XMin ,"
 y ",YMin)

#Cordenadas del máximo valor para la suma de filas y columnas
posmaxx=np.where(sumcol == XMax)
posmaxy=np.where(sumfil == YMax)
posminx=np.where(sumcol == XMin)
posminy=np.where(sumfil == YMin)

#Gráficas
print("Coordenadas para el valor maximo (", posmaxx[0][0],",",posmaxy[0][0],
")")#columna,fila
print("Coordenadas para el valor minimo (", posminx[0][0],",",posminy[0][0],
")")
```

```
Gráfica de suma de filas:
```




```
Gráfica de suma de columnas:
```

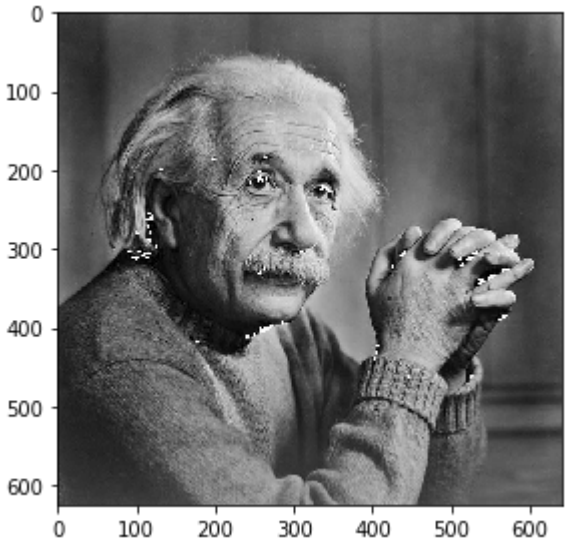



```
El valor máximo de las columnas y filas respectivamente es:  90751  y  77371
El valor mínimo de las columnas y filas respectivamente es:  34631  y  48154
Coordenadas para el valor maximo ( 268 , 283 )
Coordenadas para el valor minimo ( 36 , 392 )
```

Para cada valor calculado anteriormente, pintar dentro de la imagen una cruz de 5 píxeles x 5 píxeles. Esa cruz deberá de ser como un «+», no como una «x». El centro de cada cruz será el mínimo y el máximo. El color será blanco.

In [3]:
```
#Valores máximos y mínimos totales de color de la imagen
VMax=np.max(imtest1)
VMin=np.min(imtest1)

#Coordenadas de estos mínimos y máximos valores de la imagen
posValMax=np.where(imtest1 ==VMax)
posValMin=np.where(imtest1 ==VMin)

#Inicialización de la variable l
l=0
#Recorremos con l el número de puntos en la imagen con valor máximo
while l <(len(posValMax[0])):
    #Cruces blancas
    for i in range(5):
        imtest1[posValMax[0][l]-2+i,posValMax[1][l]]=255
        imtest1[posValMax[0][l],posValMax[1][l]-2+i]=255
    l=l+1

#Recorremos con l el número de puntos en la imagen con valor mínimo
while l <(len(posValMin[0])):
    #Cruces blancas
    for i in range(5):
        imtest1[posValMin[0][l]-2+i,posValMin[1][l]]=255
        imtest1[posValMin[0][l],posValMin[1][l]-2+i]=255
    l=l+1
#Guardar imagen
io.imsave("file1.jpg", imtest1)
#Mostrar imagen guardada
imtest2=data.imread("file1.jpg")
io.imshow(imtest2)
```

Out[3]: <matplotlib.image.AxesImage at 0x13243a54860>



Por último, se pide implementar un algoritmo que rote 180° una determinada imagen. No es necesario generalizar la rotación para cada ángulo, únicamente para 180°.

In [4]:
```
#Longitud de ejes -1 para recorrerlos en bucle for
h=625-1
w=640-1

#Array de ceros
empty_imtest1 = np.zeros([h,w], dtype=np.uint8)

#Giro de la imagen 180º
for i in range(h):
    for j in range(w):
        empty_imtest1[i,j] = imtest1[h-i,w-j]

#Guardar imagen rotada
io.imsave("filerotated.jpg", empty_imtest1)
#Leer y mostrar imagen
imrotated=data.imread("filerotated.jpg")
io.imshow(imrotated)
```

Out[4]: <matplotlib.image.AxesImage at 0x13244a92e48>