## Informe taller grupal de programación

En este taller podemos encontrar inicialmente dos puntos, donde se nos pide en el caso del primer punto crear un algoritmo, que cree una imagen con alto y ancho mostrando una diagonal de lado a lado que su parte de arriba (triangulo superior) sea de color blanco y su parte de abajo (triangulo inferior) sea de color negro. El segundo punto nos indica que utilicemos una imagen de la librería skimage, a la cual debemos implementar algún proceso como: Suavizar y resaltar contornos, Restauración, Eliminación de ruido, Mejoramiento del contraste, Detección de bordes.

Para el primer punto es necesario importar ciertas librerías para que el programa funcione

```
import numpy as np  # Importaci
from skimage import data # Impor
import matplotlib.pyplot as plt
%matplotlib inline
import sys
```

como el problema no especifica el tamaño de la imagen decidimos asignar una variable que tome el valor 100 para guardar el tamaño de la matriz, además de usar otra variable que almacene las listas que terminaran formando la imagen deseada.

```
ancho=100 # variable que guarda el tamaño de la matriz
imagen = [] # Inicializacion de una primer lista.
```

después de la asignación decidimos hacer dos ciclo for anidados, el primero recorre la variable ancho de 0 hasta 100, a su vez se asigna nombre a otra variable de lista para el siguiente for que vuelve a recorrer la variable ancho de 0 a 100, estos dos generan una matriz cuadrada,

```
for i in range(0,ancho): #
  fila = [] # Inicializaci
  for j in range(0,ancho):
```

esto es para generar la matriz ahora es requerido asignar los colores para formar la diagonal que se desea, para esto decidimos utilizar un ciclo if  donde definimos lo que pasa si las variables que realizan el recorrido por el rango (i y j) una es menor que la otra, aquí en esta parte asignamos los colores respectivos (rojo, verde y azul) para la parte de arriba todas van a tomar un valor de 255 para lograr el color blanco deseado, además de utilizar una lista llamada pixel que almacene los colores, si esto no pasa la variable i es mayor que j y aquí hacemos lo mismo que lo anterior con la diferencia que asignamos un valor de 0 para lograr ese color negro.

```
if i<j: # Condicional que est
  rojo = 255 # Variable de co
  verde = 255 # Variable de c
  azul = 255 # Variable de co
  pixel = [rojo, verde, azul]
else: # Condicional alterno q
  rojo = 0
  verde = 0
  azul = 0
  pixel = [rojo, verde, azul]
```

Posteriormente a las listas creadas les ponemos un .append para que se almacenen a modo de matrices, ejecutamos los comandos finales como print de las dimensiones, la imagen y el número de elementos.

```
    fila.append(pixel)   # Se llena 100 veces de triadas para
  #print("Dimension de fila: ",np.shape(fila))
  #sys.exit()
  imagen.append(fila) # fila es una matriz bidimencional (100*
imagen_blanca=np.array(imagen) #
plt.imshow(imagen_blanca) # Se muestra la imagen (imagen_blanc
print(len(imagen_blanca[0]))
print("Dimensiones de la imagen: ",imagen_blanca.shape)
print("Numero de elementos de la matriz: ",imagen_blanca.size)
```

En el segundo ejercicio importamos las librerías requeridas para mostrar una imagen.

```
import matplotlib.pyplot as plt#se i
from skimage import data#se importa
from skimage.color import rgb2gray#s
from skimage import feature
```

utilizamos una imagen de astronauta, aquí se ejecutan iniciando por uno que almacene la imagen del astronauta de data, otro que la modifique en este caso la iluminancia, asignándole este cambio a otra variable, finalmente se modificada por un filtro, finalmente se le dice al programa que nos muestre la imagen.

```
original=data.astronaut()#
o2= rgb2gray(original)#com
edge=feature.canny(o2)#se
plt.imshow(edge)#me muestr
plt.show()
```

Finalmente el programa es ejecutado por el colab mostrando las dos impresiones correspondientes una con la imagen cuadrada donde una diagonal parte la figura en dos colores, junto con los respectivos datos de la imagen y la otra nos muestra la imagen de la astronauta modificada.

```
100
Dimensiones de la imagen:  (100, 100, 3)
Numero de elementos de la matriz:  30000
```