Ejercicio 1

Pasar a escala de grises el color codificado en los elementos de la lista `pixel`

In [1]:
```
import statistics
```

In [2]:
```
pixel= [0.6,0.3,0.4] # Intensidades de cada canal
#El elemento 0 es el R, el 1 el G y el 2 el B

#la intensidad en escala de grises es el promedio de la intensidad de cada canal R,
intensidad=0 # IMPLEMENTAR

intensidad = statistics.mean(pixel)
print("intensidad=",intensidad)
```

```
intensidad= 0.43333333333333335
```

Ejercicio 2

Pasar a blanco y negro el valor de intensidad codificado en la variable intensidad

In [3]:
```
if (intensidad >= 0.5):
    print ("Color Blanco")
elif (intensidad <= 4.9):
    print ("Color negro")
```

```
Color negro
```

Ejercicio 3

Escribir un for para buscar el máximo de la lista e imprimirlo

In [4]:
```
lista=[44,11,15,29,53,12,30]
maximo=lista[0]

for x in range(1,6):
    if lista[x] > maximo:
        maximo = lista[x]
print ("Numero mayor=",maximo)
```

```
Numero mayor= 53
```

Ejercicio 4

Escribir un for para buscar el minimo elemento de la lista e imprimir su *posición*

In [5]:
```
lista=[44,11,15,29,53,12,30]
maximo = lista[0]
posicion=0

for x in range(1,6):
    if lista[x] > maximo:
        posicion = str(x)
        print ("Posicion=",posicion)
```

```
Posicion= 4
```

Ejercicio 5

Ordenar la lista de forma asendente

In [6]:
```
lista=[44,11,15,29,53,12,30]
ordenada = sorted(lista)
print ("Ascendente=",ordenada)
```

```
Ascendente= [11, 12, 15, 29, 30, 44, 53]
```

Ejercicio 7

Crear una funcion en donde me permita enviar como parametro el numero de elementos y devolver un listado de la serie fibonacci con el numero de elementos ingresado.

In [7]:
```
def serieFibonacci(elementos):
    a = 0
    b = 1
    while a < elementos:
        print (a, end="  ")
        a, b = b, a+b

serieFibonacci(1000)
```

```
0  1  1  2  3  5  8  13  21  34  55  89  144  233  377  610  987
```

Ejercicio 8

Escribir una función que reciba una lista y un valor, y devuelva la cantidad de veces que aparece ese valor en la lista

In [8]:
```
def calcularValor(lista, valor):
    cantidad=[]
    cont = 0
    for x in range(0,len(lista)):
        if lista[x] == valor:
            cont = cont +1
```

```
    print ("Se repiten =",cont,"veces")

l=[1,4,2,3,5,1,4,2,3,6,1,7,1,3,5,1,1,5,3,2]

calcularValor(l, 2)
```

```
Se repiten = 3 veces
```

Ejercicio 9

Investigar las funciones que se pueden utilizar con listas, diccionarios y tuplas.

Listas:

Append() permite agregar nuevos elementos

Extend() permite agregar elementos dentro de una lista, pero a diferencia de append al momento de agregar una lista, cada elemento de esta lista se agrega como un elemento más dentro de la otra lista.

Remove() va a remover un elemento que se le pase como parámentro de la lista

Index() devuelve el número de indice del elemento

Count() Para saber cuántas veces un elemento de una lista se repite

Reverse() podemos invertir los elementos de una lista

Diccionarios:

dict () Recibe como parámetro una representación de un diccionario y si es factible, devuelve un diccionario de datos.

zip() Recibe como parámetro dos elementos iterables, ya sea una cadena, una lista o una tupla.

items() Devuelve una lista de tuplas, cada tupla se compone de dos elementos

keys() Retorna una lista de elementos, los cuales serán las claves de nuestro diccionario.

values() Retorna una lista de elementos, que serán los valores de nuestro diccionario

clear() Elimina todos los ítems del diccionario dejándolo vacío.

copy() Retorna una copia del diccionario original.

fromkeys() Recibe como parámetros un iterable y un valor, devolviendo un diccionario que contiene como claves los elementos del iterable con el mismo valor ingresado.

get() ecibe como parámetro una clave, devuelve el valor de la clave. Si no lo encuentra, devuelve un objeto none.

pop() Recibe como parámetro una clave, elimina esta y devuelve su valor. Si no lo encuentra, devuelve error.

setdefault() Funciona de dos formas. En la primera como get

update() Recibe como parámetro otro diccionario. Si se tienen claves iguales, actualiza el valor de la clave repetida; si no hay claves iguales, este par clave-valor es agregado al diccionario.

Tuplas

count() Devuelve el número de veces que ocurre un valor especificado en una tupla.

index() Busca en la tupla un valor especificado y devuelve la posición de donde se encontró.