Ayudantia.md

Ayudantía I/O - Bytes

Ejercicio 1:

Pasar una imagen a un arreglo de bytes. Completar el siguiente método:

```
def file_to_bytes(path):
    pass
```

Ejercicio 2:

Crear un método que tome un arreglo de bytes en *little endian* y calcule el número en decimal:

```
def to_int(databytes):
    pass

# Probemos con nueve en hexadecimal
print(to_int(b'\x09'))  # 0xFF = 0*(16^1) + 9*(16^0) = 9

# En hexadecimal, A vale 10, B vale 11, C vale 12, ..., F vale 15.
print(to_int(b'\xFF'))  # 0xFF = 15*(16^1) + 15*(16^0) = 255

# En little endian el más a la derecha es el más significativo
print(to_int(b'\xFF\x01'))  # 0x01 0xFF = 1*(16^2) + 15*(16^1) + 15*(16^0) = 511
```

Ejercicio 3:

Haga un método que extraiga la siguiente metadata de un archivo `.bmp` en números decimales:

- Ancho
- Alto
- Tamaño
- Inicio de la data de la imagen en sí

> Puede conseguir la arquitectura del formato aquí: http://es.wikipedia.org/wiki/Windows_bitmap

```
def metadata(data):
    metadata = {}
    metadata['Tamano'] = 0
    metadata['Ancho'] = 0
    metadata['Alto'] = 0
    metadata['Inicio'] = 0
    return metadata
```

Ejercicio 4:

Cree un método que reciba el *ancho* actual de la cantidad de bytes por fila y retorne el *padding* de la imagen. Recuerde que un pixel son **3 bytes**.

```
def get_padding(pixel_width):
```

```
    pass
```

Ejercicio 5

Cree un método que remueva **todos los tonos verdes**. Este debe recibir una ruta con la imagen original y otra ruta con la ruta de salida.

```
def remove_green(source, output):
    pass
```

- ¿Por qué el fondo de la imagen ahora es **negro**?
- ¿Por qué lo que antes era blanco ahora es **purpura**?