Práctica de Katas de Programación

Debes de crear la función `process_matrix`.

Función process_matrix

Recibe como parámetro una matriz (lista de listas) de números y devuelve otra, con el mismo tamaño y número de elementos.

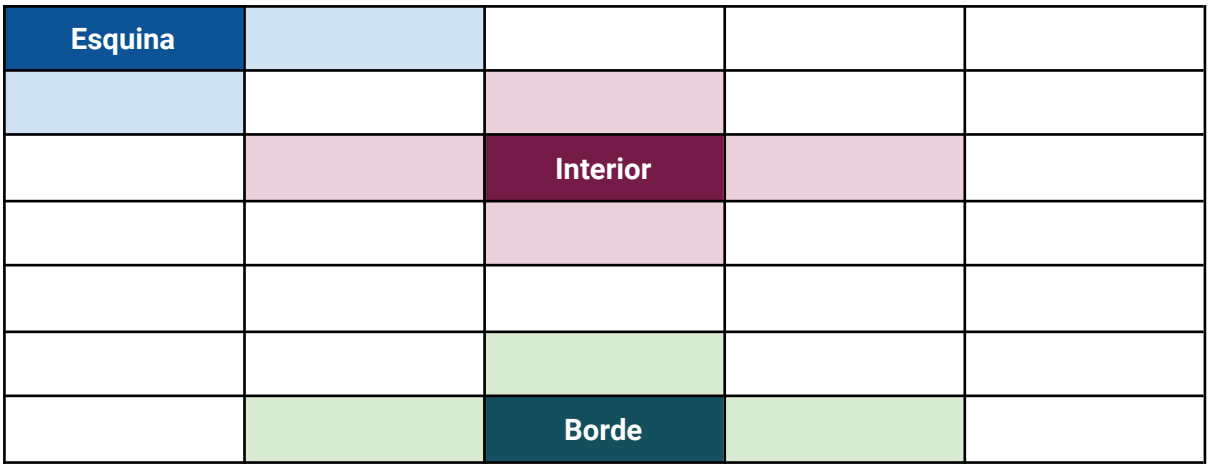
`process_matrix` transforma los elementos de la matriz original.

Transformación

La transformación que aplica es la siguiente: **cada elemento de la matriz pasa a tener como valor el promedio de su valor antiguo y los valores de sus vecinos**.

Vecinos

Se considera vecino, aquel elemento con el se comparte una arista.



Tipos de elementos

- Los elementos que están en el interior, tienen 4 vecinos. Son el caso general.
- Los elementos que se encuentran en el borde, tienen 3 vecinos. Son una excepción.

- Los elementos que se encuentran en las esquinas, tienen 2 vecinos. Son otra excepción.

Promedio

Antes

4	5			
6		0		
	-1	**7**	8	
		9		
		2		
	2	**5**	1	

Después

(4 + 5 + 6) / 3 = 5				
		(7 -1 + 0 +8 + 9) / 5 = 4.6		
		(5 + 2 + 2 +1) / 4 = 2.5		

Esto mismo se debe de aplicar a todos y cada uno de los elementos de la matriz.

Proceso de Solución

Está claro que deberemos iterar por todos los elementos de la matriz (lista de listas).

Índices

Para acceder a cada elemento, usaremos dos índices (i y j), siendo que i representa la columna y j la fila.

0,0	1,0	2,0
0,1	1,1	2,1
0,2	1,2	2,2

Para recorrer todos los elementos, vamos a necesitar un bucle anidado (un for dentro del otro):

```
for i, column in enumerate(matrix):
      for j, value in enumerate(column):
            new_value = process_element(....)
```

process_element

Vista la estructura general, procesar una lista y devolver otra, cosa que ya hemos visto mil veces, podemos centrarnos en la transformación que vamos a aplicar a cada uno de los elementos.

La función `process_element` va a recibir los índices del elemento y la matriz original, y devolverá el nuevo valor.

Divide y Vencerás

A primera vista, la tarea se puede descomponer en las siguientes subtareas:

- Descubrir los vecinos de un elemento
- Obtener su promedio
- Promediarlo con el valor antiguo

El principal escollo parece ser el hecho de que según cual sea el elemento, el número de vecinos cambia. Da la impresión de que tendremos tres reglas de vecindario: interior, borde y esquina.

Versión Reducida

Para poder empezar, vamos a resolver una versión reducida del problema, antes de ponernos con la versión completa. En vez de una matriz, vamos a procesar una lista.

La función `process_list` hace lo mismo que la función `process_matrix`, pero actúa sobre una lista de números.

Esto simplifica el bucle principal y sobre todo simplifica la tarea de encontrar vecinos. Ahora solo hay dos tipos (en vez de tres):

- Interior: tiene dos vecinos
- Borde: el primero y el último sólo tienen 1 vecino

Inicial	Interno	Interno	Interno	Final

Una vez tengas resuelta la versión reducida, **puedes ya abordar la versión completa**.

Haremos hasta aquí juntos.

Entrega

Deberás subir a una carpeta de google drive que se te indicará, un fichero de python con la función `process_matrix`.

El nombre de dicho fichero debe de ser tu nombre y apellidos, separados por guiones bajos, y terminado en .py.

Por ejemplo: `ana_perez_blanco.py`

¿Qué hemos creado?

El proceso de transformar cada punto en el promedio de sus vecinos es un filtro de imágenes muy común. La gran diferencia está en el tamaño de las imágenes, representadas como matrices de píxeles (3 números que representan los componentes rojo, verde y azul), son mucho mayores que las que hemos visto aquí.

Al ir promediando los valores, lo que hacemos es eliminar detalles y bordes, es decir, difuminamos la imágen. Has creado **un filtro para eliminar poros abiertos, arrugas y patas de gallo en el instagram**.

Si en vez de procesar cada elemento, lo hubiésemos hecho *por bloques*, en vez de suavizar, estaríamos *pixelando* la imagen. Si además, solo lo aplicas a un rango de índices (en vez de toda la matriz), habríamos pixelado *una parte de la imagen*, como la cara.