ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN
CCPG1001 - FUNDAMENTOS DE PROGRAMACIÓN
SEGUNDA EVALUACIÓN - II TÉRMINO 2018-2019/ Febrero 1, 2019

Nombre: _____ Matrícula: _____ Paralelo:____

COMPROMISO DE HONOR: Al firmar este compromiso, reconozco que el presente examen está diseñado para ser resuelto de manera individual, que puedo usar un lápiz o esferográfico; que sólo puedo comunicarme con la persona responsable de la recepción del examen; y, cualquier instrumento de comunicación que hubiere traído, debo apagarlo y depositarlo en la parte anterior del aula, junto con algún otro material que se encuentre acompañándolo. Además no debo usar calculadora alguna, consultar libros, notas, ni apuntes adicionales a los que se entreguen en esta evaluación. Los temas debo desarrollarlos de manera ordenada.
Firmo el presente compromiso, como constancia de haber leído y aceptado la declaración anterior. "Como estudiante de ESPOL me comprometo a combatir la mediocridad y actuar con honestidad, por eso no copio ni dejo copiar".

_____
Firma

La World Wide Fund for Nature (WWF) se encuentra realizando el #TenYearChallenge del Ártico. Para esto tiene información sobre la cantidad de hielo y población de diversas especies durante los años 2009 y 2019 en un archivo con el siguiente formato:
Número de filas
Número de columnas
Año,fila,columna,hielo,especie

Ejemplo:

```
22
14
Año,fila,columna,hielo(0/1),especie
...
2009,14,8,1,9
...
2019,5,7,0,3
```

Implemente las siguientes funciones:

1. **[15 puntos] crearMatriz(nomArchivo)** que recibe el nombre del archivo con los datos de la WWF y retorna una tupla que contiene las siguientes cuatro matrices:

Cantidad de hielo

2009

1	0	0	1	...
0	1	1	1	...
1	0	1	0	...
1	1	1	0	...
0	1	0	1	...
1	1	1	0	...
...	...	..	...	...

2019

1	0	0	1	...
0	1	0	1	...
1	0	1	0	...
1	0	1	0	...
0	1	0	1	...
0	0	1	0	...
...	...	...	...	...

Especies de animales

2009

1	2	4	4	...
2	5	5	3	...
1	3	9	1	...
1	1	4	2	...
4	22	4	7	...
1	1	14	4	...
...	...	..	...	...

2019

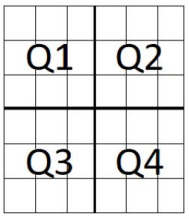
3	2	4	1	...
0	11	3	5	...
1	0	67	1	...
2	22	3	2	...
13	13	0	2	...
3	0	2	3	...
...	...	...	...	...

En las matrices de hielo cada valor uno (1) en una celda representa la presencia de hielo. En las matrices de especies cada celda representa la presencia de un animal de una especie dependiendo de una cierta codificación, ejemplo:

0: No hay animal 1: Lobo ártico 2: Oso Polar 3: Reno 4: Foca 5: ...

Para el resto del examen considere lo siguiente:

Las matrices pueden ser divididas en cuadrantes (**cuatro subregiones de igual tamaño**) llamados Q1, Q2, Q3 y Q4, como se muestra abajo:



Asuma que existen las siguientes funciones:

A. **cuadrantes(matriz)** que recibe una matriz y devuelve una tupla con los cuadrantes de la matriz, es decir, retorna (Q1, Q2, Q3, Q4) que son a su vez matrices de Numpy.
B. **poblacionEspecie(mAnimales, especie)** que recibe una matriz de animales y el código de una especie. La función retorna una tupla con la cantidad total de animales de esa especie para cada uno de los cuadrantes. La tupla tendrá el siguiente formato (pQ1, pQ2, pQ3, pQ4).

2. **[10 puntos] densidadHielo(mHielo)** que recibe una matriz de hielo y retorna una tupla con la densidad de hielo para cada cuadrante. La densidad de un cuadrante es la cantidad total de sus celdas con valor uno (1) dividido para el número total de celdas del cuadrante. La tupla tendrá el siguiente formato (dQ1, dQ2, dQ3, dQ4)
3. **[20 puntos] especieDominante(mAnimales)** que recibe una matriz de animales y retorna una tupla con los códigos de la especie que más se repite (dominante) en cada cuadrante. La tupla tendrá el siguiente formato (eQ1, eQ2, eQ3, eQ4).
4. **[15 puntos] migracionEspecie(mAnimales2009, mAnimales2019, especie)** que recibe la matriz de animales del 2009, la matriz de animales del 2019 y el código de una especie. La función debe retornar una tupla con dos strings. El primer valor corresponde al cuadrante (`'Q1'`, `'Q2'`, `'Q3'` o `'Q4'`) donde hubo mayor población de animales de la especie en el 2009. El segundo valor corresponde al cuadrante donde hay mayor población de animales de la especie en el 2019.

Para el siguiente numeral, asuma que tiene un diccionario con los nombres de **todas** las especies del ártico en el siguiente formato:

dicEspecies = {codigo_especie:nombre_especie}

5. **[20 puntos] crearDiccionario(mHielo, mAnimales, dicEspecies)** que recibe una matriz de hielo, una matriz de animales y el diccionario de las especies del ártico. La función retornará un diccionario similar al mostrado abajo:

```
dic = {'densidad hielo':{'Q1':0.75, 'Q2':0.11, 'Q3':0.55, 'Q4':0.47},
       'Especies':{ 1:51,  #{especie:poblaciónTotal}
                    2:6,
                    ...
                    67:93}
      }
```

Luego, implemente un programa principal que:

6. **[1 punto]** Forme las matrices a partir del archivo artico2009-2019.txt
7. **[9 puntos]** Muestre por pantalla el código de las especies que en el 2009 son comunes (se repiten) para los cuadrantes Q1, Q3 y Q4 pero que no existan en Q2.

Los siguientes dos numerales se resuelven juntos.

8. **[1 punto]** Determine si ha habido migración de la especie con código 57. **Existe migración si el cuadrante con mayor población en el 2009 y 2019 son diferentes**.
9. **[9 puntos]** Una posible explicación para la migración de una especie puede ser el deshielo. Entonces, si existió migración (de la especie con código 57) del cuadrante con mayor población en el 2009 (Qx) al cuadrante con mayor población en el 2019 (Qy), muestre la diferencia (positiva o negativa) de densidad de hielo del 2009 al 2019 para el cuadrante Qy. Calcule la diferencia con la siguiente fórmula:

Diferencia = Qy_2019 - Qy_2009

Por ejemplo:

Si `migracionEspecie` retorna ('Q1', 'Q4'), la diferencia será la densidad de hielo en el 2009 del cuadrante 'Q4' menos la densidad de hielo en el 2019 del mismo cuadrante 'Q4'.

Tema extra por puntos adicionales en la siguiente página.

BONO (+10 PUNTOS)
¿Qué imprime el siguiente código? Justifique su respuesta

```
import numpy as np
M = np.ones((20,17), int)
M1 = M[5:14,7:15]
M2 = M[1:9,8:]
f,c = M2.shape
if M1.size == f*c:
     M[5:14,7:15] = 0
else:
     M[1:9,8:] = -1
print(M.sum())
```

Asuma que este tema NO tiene errores de compilación. Si usted cree que hay algún error de compilación, consúltelo inmediatamente con su profesor.

---//---

Cheat Sheet. Funciones y propiedades de referencia en Python.

Librería Numpy para ***arreglos*:**	para ***listas***:	para ***cadenas***:
np.array([elementos],dtype=) np.unique(**arreglo**) np.sum(***arreglo***) np.mean(***arreglo***) ***arreglo***.shape ***arreglo***.size ***arreglo***.sum()	***listas***.append(...) ***listas***.extend(...) ***listas***.count(...) ***listas***.index(...) ***listas***.pop() *elemento* in ***listas***	***cadenas***.islower() ***cadenas***.isupper() ***cadenas***.lower() ***cadenas***.upper() ***cadenas***.split(...) ***cadenas***.find(...) ***cadenas***.count(...)