ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN
CCPG1001 - FUNDAMENTOS DE PROGRAMACIÓN
TERCERA EVALUACIÓN - II TÉRMINO 2017-2018/ Febrero 23, 2018

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

TEMA 1 (40 PUNTOS)

CuidaVehículo ha lanzado una aplicación para llevar un control sobre mantenimiento de tu vehículo. Para esto, tienen un diccionario de mantenimientos preventivos y correctivos recomendados para cada parte del carro de acuerdo al número de **días** y **kilómetros**.

```
dictMantenimiento = {
     'preventivo': [ ('llantas', 60, 4500), … , ('bujías', 45, 3000) ],
     'correctivo': [ ('llantas', 90, 6000) , … , ('zapatas', 120, 10000) ]
}
```

Además, posee un historial por vehículo en donde se registra el número de días y el kilometraje que han pasado desde el último mantenimiento.

```
dictHistorial = {
     'GEC-2411': {
           'propietario': 'Eduardo Cueva',
           'registro': [ ('llantas', 12, 325), … ('zapatas', 180, 500) ]
     },
     …
     'GAA-0321': {
           'propietario': 'Andrea Martínez',
           'registro': [ ('bujías', 40, 500), … ('zapatas', 120, 100) ]
     }
}
```

Desarrolle:

1. [15 puntos] La función **mantenimientos(strPlaca, dictHistorial, dictMantenimiento)** que recibe la placa, el diccionario con el historial por vehículo y el diccionario de mantenimientos. Esta función devolverá una lista de tuplas. Cada tupla tendrá el nombre de la parte del carro y el tipo de mantenimiento a realizar ('preventivo', 'correctivo' o 'nada') para dicha placa.

   Considere que sobre cada parte del carro solo se puede realizar un tipo de mantenimiento, preventivo o correctivo, de acuerdo a si supera el número de kilómetros. En caso de que el número de kilómetros supere el establecido para ambos tipos de mantenimiento, se deberá **preferir el mantenimiento correctivo**.

2. [15 puntos] La función **semaforo(strPlaca, dictHistorial, dictMantenimiento)** que recibe la placa, el diccionario con el historial por vehículo y diccionario de mantenimientos. Esta función devuelve un diccionario cuyas claves son 'amarillo', 'verde' y 'rojo'. Los valores corresponden a listas de partes para esa placa que necesitan un mantenimiento preventivo (en la clave 'amarillo'), correctivo (en la clave 'rojo') o ningún mantenimiento (en la clave 'verde')

3. [10 puntos] La función **recomendar(dictHistorial, dictMantenimiento, strParte, strTipoMantenimiento)** que recibe el diccionario con el historial por vehículo, diccionario de mantenimientos, el nombre de una parte, y el tipo de mantenimiento (“preventivo” o “correctivo”). La función deberá retornar una lista con todas las placas de los vehículos que ya necesitan mantenimiento de **strTipoMantenimiento** en **strParte**.

TEMA 2 (50 PUNTOS)

Suponga que tiene el archivo “videojuegos.csv” con información sobre todo el contenido de su biblioteca de videojuegos. El archivo tiene la siguiente estructura:

videojuegos.csv

```
Nombre,Año,Consola,Calificación,Tags(separados por ;)
The Legend of Zelda,86,Famicom Disk System,3.5,RPG;Link;Zelda;Hyrule;Triforce
Double Dragon,87,Arcade,3.7,Beat-'em up;Billy;Jimmy;Puñete
The Legend of Zelda,88,NES,4.3,RPG;Link;Zelda;Hyrule;Triforce
…
Halo 5: Guardians,15,Xbox One,4,FPS;Master Chief;Cortana;Covenant
```

Note que un juego aparecerá listado en el archivo una vez por cada consola en la que fue lanzado.
La categoría del juego se especifica siempre en el primer Tag. Por ejemplo, Double Dragon pertenece a la categoría Beat-’em up.

Desarrolle lo siguiente:

1. [15 puntos] La función **juegosConsolas(nomArchivo, categoria, decada)** que recibe el nombre del archivo con la información de los videojuegos, una categoría y un número de cuatro dígitos representando una década de años. La función retorna una tupla con 2 elementos. El primer elemento es la lista con valores únicos de todos los juegos de esa década para esa categoría. El segundo elemento es la lista con valores únicos de todas las consolas que tienen juegos para esa década y categoría.

   Por ejemplo, llamar a `juegosConsolas('videojuegos.csv', 1980, 'RPG')` retorna:
   `(['The Legend of Zelda', 'Phantasy Star', ...],['NES', 'Famicon Disk System', ...])`

2. [17 puntos] La función **crearMatriz(nomArchivo, categoria, decada)** que recibe el nombre del archivo con la información de los videojuegos, el nombre de una categoría de videojuegos y un número de cuatro dígitos representando una década de años. La función deberá leer el archivo y retornar una matriz donde las filas representan los juegos de **categoria** para la **decada**, las columnas representan las consolas que tienen juegos de **categoria** para la **decada** y las celdas son las calificaciones de cada juego para cada consola. Si un juego no existe para una consola, su calificación deberá ser cero (0).

3. [18 puntos] La función **mejoresJuegos(nomArchivo, categoria, decada)** que recibe el nombre del archivo con la información de los videojuegos, el nombre de una categoría de videojuegos y un número de cuatro dígitos representando una década de años. La función deberá generar el archivo *“Mejores.txt”* con las cinco mejores consolas de la **decada** para la **categoria**, ordenados de mayor a menor por su **calificación promedio**. *Para calcular el promedio de una consola, considere únicamente los juegos que fueron lanzados para dicha consola (no considere los valores cero)*. El archivo tendrá la siguiente estructura:
   `Consola,Promedio_calificación`

4. [*Bono: 5 puntos*] La función **colecciones(nomArchivo, palabras)** que recibe el nombre del archivo con la información de los videojuegos y una lista de palabras. La función deberá retornar otra lista de valores únicos con los nombres de todos los juegos que sus Tags contengan **todos** los términos de la lista ***palabras***.

TEMA 3 (10 PUNTOS)

1. Considere el siguiente código

```
a = 'ABACDEF'
b = [4,2,1,3,2,2,3]
d = ''
for i in a:
   if i not in d:
       d += i * b[ list(a).index(i)]
   else:
       d = d.replace(i, '')

print(d)
```

2. Indique la salida por pantalla del siguiente código. Justifique su respuesta.

```
a = [1,3,5,6,8]
b = []
for i in range(len(a)):
    b.append(i)
    b.insert(i,len(b))
print(b)
```

---//---

Cheat Sheet. Funciones y propiedades de referencia en Python.

Librería Numpy para ***arreglos*:**	para ***listas***:	para ***cadenas***:
np.array((numRows,numCols),dtype=) ***arreglos***.shape ***arreglos***.reshape() np.sum(***arreglos***) np.mean(***arreglos***) np.where(**condición**) np.argsort(**arreglo**) np.unique(**arreglo**) ***arreglos***.sum(axis=1)	***listas***.append(...) ***listas***.count(...) ***listas***.index(...) ***listas***.pop() *elemento* in ***listas***	***cadenas***.islower() ***cadenas***.isupper() ***cadenas***.lower() ***cadenas***.upper() ***cadenas***.split(...) ***cadenas***.find(...) ***cadenas***.count(...)