ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN
CCPG1001 - FUNDAMENTOS DE PROGRAMACIÓN
SEGUNDA EVALUACIÓN - I TÉRMINO 2017-2018/ Septiembre 2, 2017

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)

Se necesita crear un programa para generar las estadísticas de las palabras en un texto. Por ejemplo, a partir del siguiente texto:

```
Con,la,ayuda,de,un,grupo,de,amigos,y,de,valientes,aliados,Frodo,emprende,un,peligroso,viaje,con,la,mis
ión
de,destruir,el,Anillo,Único,Pero,el,Señor,Oscuro,Sauron,quien,creara,el,Anillo,envía,a,sus,servidores
para,perseguir,al,grupo,Si,Sauron,lograra,recuperar,el,Anillo,sería,el,final,de,la,Tierra,Media,Ganado
ra
de,cuatro,Oscars,este,inmortal,relato,sobre,el,bien,y,el,mal,la,amistad,y,el,sacrificio,te,transportar
á
a,un,mundo,más,allá,de,tu,imaginación
...
```

su programa generaría el siguiente diccionario, apoyándose en las cinco funciones que se detallan abajo. Asuma que **cada palabra tendrá 20 caracteres como máximo** y que **el número máximo de palabras por líneas es 30.** También tiene una lista **stopwords** = ['la', 'con', ...] de palabras que no agregan mayor significado al texto:

```
{
   'NTP': 83, 'NPC': 22,
   'palabras':
   {
     'Anillo': {'veces': 3, 'NL': (2,3,4)},
     'Sauron': {'veces': 2, 'NL': (2,3)  },
               ...
   }
}
```

Donde,

NTP = Número total de palabras
NPC = Número solo de palabras comunes
NL = Número de líneas

Ahora implemente:

1) La función **cargarArchivo(nombre),** que leerá el texto desde el archivo **nombre** y creará una **matriz M de NumPy (con dtype='U20')** donde cada fila representa una línea, y cada columna, una palabra de dicha línea. Si la línea tiene menos de 30 palabras, las celdas restantes deben ser llenadas con un string vacío. Cada línea del archivo está limitada por **'\n'**. Cada palabra está separada por una coma. (6 puntos + 5 puntos de bono)

2) La función **ocurrencias(palabra, M),** que devuelve el número de veces que el string **palabra** aparece en la matriz **M**. (6 puntos)

3) La función **líneas(palabra, M)**, que devuelve una tupla con los números de fila donde aparece **palabra** en **M**. (6 puntos)

4) La función **contarPalabras(M, stopwords)** que devuelve una tupla con el número total de palabras del texto (**incluyendo las palabras stopwords**) y el número **solo** de palabras stopwords. **Cada palabra (regular o stopword) debe ser contada una sola vez** sin importar cuantas veces se repita en el texto. (11 puntos)

5) Implemente la función **concordancia(M, stopwords)** que devuelve un diccionario con las estadísticas de las palabras (ver ejemplo arriba). El diccionario interno `'palabras'` **no debe incluir las palabras stopwords**. (11 puntos)

TEMA 2. (50 PUNTOS)

Para administrar la nueva matriz energética del Ecuador, Ud. tiene un **diccionario con la información de las ciudades y las plantas de energía que las atienden.** Cada planta tiene: una tupla con los consumos mensuales (12) del año en megavatios-hora (MWh) y la tarifa de consumo en dólares por megavatio-hora (MWh) que le cobra a la ciudad. **Una ciudad es servida por al menos una planta eléctrica. No todas las ciudades son servidas por todas las plantas eléctricas.**

```
consumo_energia = {
                   'Quito': {
                        'Coca Codo Sinclair':  { 'consumos':(400, 432, …, 213),'tarifa': 65},
                        'Sopladora': { 'consumos': (120, 55, 32, …, 70),'tarifa': 84},
                        ...
                        },
                   'Guayaquil':  {
                        'Coca Codo Sinclair':{ 'consumos': (310, 220, 321, …, 200),'tarifa':55},
                        'Paute': { 'consumos':  (400, 432, …, 587),'tarifa': 79},
                        'Agoyán': { 'consumos': (50, 32, 32, …, 40),'tarifa': 32},
                        ...
                        },
                  ...
                  }
```

Además, dispone del siguiente diccionario con información de **planta por región**:

```
informacion = {
       'costa': ('Coca Codo Sinclair', …),
       'sierra': ('San Francisco', 'Agoyán', …),
       'oriente': ('Paute', 'Sopladora', …),
}
```

Implemente lo siguiente:

1) Una función **total_anual(consumo_energia, planta, ciudad)** que recibe el diccionario **consumo_energia**, el nombre de una planta y el nombre de una ciudad. La función debe calcular y retornar el total anual de megavatios-hora servido por **planta** a **ciudad**. (7 puntos)

2) Una función **total_plantas_ciudad(consumo_energia, planta)** que recibe el diccionario **consumo_energia** y el nombre de una planta. La función debe devolver un diccionario cuyas claves corresponden a los nombres de las ciudades a las que **planta** provee energía y los valores corresponden al total anual de megavatios-hora servido por **planta** a cada ciudad. (13 puntos)

3) Una función **megavatios_hora(consumo_energia, informacion)** que recibe el diccionario **consumo_energia** y el diccionario **informacion**. La función retorna el total anual de megavatios-hora generados por las plantas de energía de la región SIERRA. (15 puntos)

4) Una función **facturacion(consumo_energia)** que recibe el diccionario **consumo_energia** y genera un archivo con la facturación total en dólares de los seis últimos meses de cada ciudad. El archivo resultante se llamará facturacion.txt y tendrá la siguiente estructura: (15 puntos)

```
Ciudad,julio,agosto,septiembre,...,diciembre
Guayaquil,2903,2145,3010,...,2945
Quito,3102,3234,3223,...,3417
…
```

TEMA 3 (10 PUNTOS)

a. Considere:

```
archivo = open('datos.txt','w')
archivo.write('Días de la semana\n')
archivo.close()
lista = ['lunes', '\n', 'martes', '\n', 'miércoles', 'jueves', 'viernes']
archivo = open('datos.txt','a+')
archivo.writelines(lista)
archivo.close()
```

¿Cuántas líneas de texto tiene el archivo datos.txt al final del programa y con qué contenido? Justifique su respuesta. (5 puntos)

b. Muestre la salida por pantalla de la ejecución del siguiente código y justifique su respuesta: (5 puntos)

```
a = set([3,5,6,7,8])
b = set([6,7,8,9,10])
c = set([2,4,5])
d = set([len(a), len(c), len(b)])
f = (a | b) & d
g = (b - c) | d
h = d ^ a
print(f)
print(g)
print(h)
```

---//---

Cheat Sheet. Funciones y propiedades de referencia en Python.

Librería Numpy para ***arreglos*:**	para ***listas***:	para ***cadenas***:
np.array((numRows,numCols),dtype=) np.empty((numRows,numCols),dtype=) ***arreglos***.shape ***arreglos***.reshape() numpy.sum(***arreglos***) numpy.mean(***arreglos***) ***arreglos***.sum(axis=1) ***arreglos***.fill(valor)	***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(...)