IWI131 Programación (Fase 2) — Ejercicios Semana 3: Diccionarios y Archivos

1. En el año 28908 hay elecciones municipales en Pythonlandia y cada mesa de votación cuenta con una urna sumamente tecnológica. Esta urna escanea el voto depositado y escribe la información en un archivo siguiendo el formato `rut:voto`, donde `rut` es el rut de la persona y `voto` es el código del candidato por el cual vota dicha persona. Ver ejemplo: `votaciones.txt`. Además, el código y nombre de los candidatos se encuentran en el archivo `candidatos.txt`:

`votaciones.txt`

```
9173315-3:A1
19642628-0:A2
21933617-9:X27
173315-8:A2
516314-1:C2
...
```

`candidatos.txt`

```
A1:Alejandro Pinera
A2:Sebastian Guillier
A3:Beatriz Kast
A4:Leopoldo Sharp
A5:Jorge Mendez
...
```

Desarrolle la función `resultados(vot, can)`, donde `vot` y `can` son strings con los nombres de los archivos anteriores. Esta función debe generar el archivo `resultados.txt`, cuyo encabezado es el total de votos válidamente emitidos, es decir, eliminando los votos nulos. Luego está el porcentaje de votos que obtuvo cada candidato ordenado de mayor a menor. Si un elector vota por un candidato que no se encuentra en el archivo `candidatos.txt` el voto es nulo. Guíese por el ejemplo (`resultados.txt`):

```
>>> resultados('votaciones.txt', 'candidatos.txt')
```

`resultados.txt`

```
Total de votos: 3

Sebastian Guillier 66.6%
Alejandro Pinera 33.3%
Beatriz Kast 0.0%
Leopoldo Sharp 0.0%
Jorge Mendez 0.0%
```

IWI131 Programación (Fase 2) — Ejercicios Semana 3: Diccionarios y Archivos

2. Un servicio de streaming le ha pedido a sus suscriptores que voten y dejen comentarios sobre sus series favoritas. Toda la información recopilada se tiene en el archivo `opiniones.txt`. Este archivo contiene en cada línea el título de la serie, el comentario de la persona y su voto.

opiniones.txt

```
Lost:Que gran final:4.8
Westworld:Interesante, pero aburre:2.5
GOT:Fome, aguante SPARTACUS:1.0
Westworld:No entiendo nada:4.5
Mr. Robot:La T2 apesta:5.0
GOT:#DominGOT:5.0
...
```

- **TAREA 1.** Escriba la función `promedio(archivo)` que recibe un string con el nombre del archivo y retorna un diccionario cuyas llaves sean el nombre de todas las series del archivo y su valor un flotante con el promedio de las opiniones de la gente.

```
>>> promedio('opiniones.txt')
{'Lost': 4.8, 'Westworld': 3.5, 'GOT': 3.0, 'Mr. Robot': 5.0}
```

- **TAREA 2.** Escriba la función `opinar(archivo, serie, comentario, nota)` que recibe un string con el nombre del archivo, un string con el título de una serie, un string con un mensaje y un flotante con un voto. Esta función debe agregar esa información al final del archivo.

```
>>> opinar('opiniones.txt', 'GOT', 'Me gusta pero asusta', 4.0)
```

opiniones.txt

```
Lost:Que gran final:4.8
Westworld:Interesante, pero aburre:2.5
GOT:Fome, aguante SPARTACUS:1.0
Westworld:No entiendo nada:4.5
Mr. Robot:La T2 apesta:5.0
GOT:#DominGOT:5.0
GOT:Me gusta pero asusta:4.0
```

IWI131 Programación (Fase 2) — Ejercicios Semana 3: Diccionarios y Archivos

3. Un concesionario de una autopista mantiene información relevante de sus clientes en 3 archivos: Un registro diario de patentes que circulan por ella, la información de las patentes que contienen o no contienen TAG (indicado con un 1 o un 0 respectivamente) y por ultimo, la información de los dueños y de sus patentes asociadas. A modo de ejemplo se presenta cada uno de los archivos en el cuadro a continuación:

`registro_diario.txt`

```
abmn32
crtj12
dflp11
hb5101
...
```

`info_dueños.txt`

```
Alex Perez;crtj12,hb5101,kcfl36,emda16
Aquiles Castro ;abmn32,tljg99,avrv33
Maria Gana;ab7677
Fede Santos;utfs90,dflp11
...
```

`patentes.txt`

```
crtj12,1
abmn32,0
hb5101,0
dflp11,1
tljg99,0
jfzo10,0
kcfl36,0
utfs90,0
ab7677,0
avrv33,0
emda16,0
...
```

De la información de ejemplo se desprende que `Alex Perez` tiene cuatro vehículos con patentes `crtj12`, `hb5101`, `kcf136` y `emda16`. Solo `crtj12` cuenta con dispositivo TAG y los otros tres no. Además, los vehículos con patente `crtj12` y `hb5101` pasaron durante el día por la autopista. Siguiendo con el ejemplo, `Maria Gana` tiene solamente un vehículo y no circuló durante el día por la autopista. Considere que en el archivo `info_dueños.txt` cada persona aparece solamente una vez y puede tener un número variable de patentes.

a) Para saber cuantos vehículos tiene cada persona construya la función `patentes_por_dueño(ar)` que recibe el nombre del archivo que contiene la información de los dueños y retorna un diccionario. Este diccionario debe tener como llave el nombre y apellido de la persona y como valor el número de vehículos que posee.

```
>>> patentes_por_dueño('info_dueños.txt')
{'Alex Perez': 4, 'Aquiles Castro ': 3, 'Fede Santos': 2, 'Maria Gana': 1}
```

b) Se requiere listar a aquellas patentes que circularon durante el día por la autopista y que no disponen de TAG. Para esto, construya la función `patentes_multadas(registro,patentes)` que recibe los nombres de los archivos que contienen el registro diario de patentes y el registro general de patentes respectivamente. Esta función debe retornar una lista con las patentes que satisfacen el criterio ya mencionado.

```
>>> patentes_multadas('registro_diario.txt', 'patentes.txt')
['abmn32', 'hb5101']
```

c) Finalmente, se necesita listar el nombre de cada persona multada. Para esto construya la función `personas_multadas(registro, patentes, dueños)` que recibe los nombres de los tres archivos anteriores. Esta función debe retornar una lista con los nombres de los dueños de las patentes multadas. Este listado no debe contener nombres repetidos.

```
>>> personas_multadas('registro_diario.txt','patentes.txt','info_dueños.txt')
['Aquiles Castro ', 'Alex Perez']
```

IWI131 Programación (Fase 2) — Ejercicios Semana 3: Diccionarios y Archivos

4. PyLab es un laboratorio que está desarrollando vacunas para dos tipos de enfermedades. Cada mes algunos pacientes son inyectados con vacunas experimentales y se mide la presencia de tres sustancias en su cuerpo: `BR1`, `BR2` y `KR`.

En un archivo (`datos.txt`) se guardan los resultados tras probar una nueva vacuna. La información de un paciente está en una línea del tipo: `CP;G;BR1;BR2;KR`, donde `CP` es el código del paciente, `G` es el género (`1`: femenino, `0`: masculino) y `BR1`, `BR2`, `KR` son las tres sustancias que, cuando están presentes, tienen un `1` y sino un `0`. Por ejemplo la paciente `JP87` solo presenta la sustancia `KR`.

datos.txt

```
JP87;1;0;0;1
AY0231;0;1;1;0
DR762399;0;0;0;0
ED235;1;0;0;1
W4350;1;0;1;1
...
```

La probabilidad de desarrollar la enfermedad 1 está dada por:

$$\textbf{Enfermedad}-1 = 10 \cdot KR + 30 \cdot BR2 + 4$$

La probabilidad de desarrollar la enfermedad 2 depende del genero y está dada por:

$$\textbf{Enfermedad}-2 = \begin{cases} 15 \cdot BR1 + 65 \cdot BR2 + 3 & \text{si } G == 1, \\ 5 \cdot BR1 + 8 \cdot BR2 + 6 & \text{si } G == 0. \end{cases}$$

Por otro lado se cuenta con el archivo histórico de pacientes en el formato `CP;PE1;PE2`, donde `CP` es el código del paciente, `PE1` es la probabilidad de desarrollar la enfermedad 1 y `PE2` la enfermedad 2.

Escriba la función `actualizar(ad,ah,ah2)`, donde `ad` es el nombre de un archivo de datos tras probar una nueva vacuna (como `datos.txt`), `ah` es el nombre de un archivo de tipo histórico (como `historico.txt`) y `ah2` es el nombre del nuevo archivo donde se escribirá la información actualizada considerando los dos archivos anteriormente descritos (el de datos y el histórico).

historico.txt

```
JP87;4 %;3 %
KL898989;4 %;14 %
AY0231;4 %;6 %
DR762399;34 %;11 %
ED235;14 %;3 %
W4350;14 %;68 %
...
```

H2.txt

```
JP87;14 %;3 %
KL898989;4 %;14 %
AY0231;34 %;19 %
DR762399;4 %;6 %
ED235;14 %;3 %
W4350;44 %;68 %
...
```

Por ejemplo al ejecutar `actualizar("datos.txt", "historico.txt", "H2.txt")`, se genera el archivo `'historico2.txt'` como se ve en el ejemplo anterior.

El archivo `ah2` será un **nuevo** archivo histórico en donde, si el paciente existía y probó la vacuna, se reemplazan las probabilidades de las enfermedades según el archivo de datos (`ad`). Si el paciente no existía, lo agrega en cualquier lugar y finalmente si el paciente existía, pero no probó la vacuna, es decir, no aparece en el archivo `ad`, lo copia directo en el archivo `ah2`. Guíese por el ejemplo.

Nota: Los archivos de ejemplo tienen mas líneas. Recuerde que puede crear todas las funciones auxiliares que estime conveniente.

IWI131 Programación (Fase 2) — Ejercicios Semana 3: Diccionarios y Archivos

5. El traductor de Openinglish.com es un éeexito porque ha recopilado en el archivo `english.txt` todas las palabras del idioma inglés y su correspondiente traducción al español.

english.txt

```
the:el
i:yo
and:y
to:a
my:mi
or:o
...
```

libro.txt

```
Call me Ishmael Some years ago never
mind how long precisely having little
or no money in my purse and nothing
particular to interest me on shore I
thought I would sail about a little
and see the watery part of the world
...
```

Ahora es posible traducir cualquier archivo de texto en inglés al español buscando la traducción de cada término en el archivo `english.txt` y escribiendo en un nuevo archivo la palabra en español.

Escriba la función `traducir(archivo)` que recibe un string con el nombre del archivo que se quiere traducir y genera un archivo de nombre `'traducido.txt'` con todas las palabras del archivo original traducidas. Si alguna palabra no aparece en el archivo `english.txt` es porque no tiene traducción y se debe mantener intacta. Vea a continuación que resultado entrega el traductor de Openinglish para el archivo `libro.txt`

traducido.txt

```
llama me ishmael algunos años atras nunca
mente como largo precisamente teniendo poco
o no dinero en mi bolso y nada
particular a interes me en costa yo
pensaba yo haria navegar sobre un poco
y ver el mojado parte de el mundo
```

IWI131 Programación (Fase 2) — Ejercicios Semana 3: Diccionarios y Archivos

6. En el cine *Ciclo* el precio de la entrada para ver cualquier película por primera vez es de $2000, luego el precio va reduciéndose a la mitad, cada vez que una misma persona ve la misma película, teniendo en cuenta que el precio más bajo a pagar por una entrada es $125.

Para esto, el cine cuenta con un archivo llamado `clientes.txt` como se muestra en el ejemplo, en el cual se registra a todos sus clientes solo con un nombre y un apellido (separado por espacio).

`clientes.txt`

```
Luca Caminacielos
Artur Ito
...
```

Por cada cliente se tiene un archivo llamado `nombre_apellido.txt`, cuyo contenido corresponde a cada película que dicha persona ha visto en este cine, donde la primera línea corresponde a la primera película que este cliente vio en este cine, luego la segunda línea corresponde a la segunda película vista y así sucesivamente. A continuación se muestra a modo de ejemplo las películas que han visto los clientes del archivo anterior.

`Luca_Caminacielos.txt`

```
Star Wars
Rogue One
Star Wars
El origen
Star Wars
Star Wars
Star Wars
Star Wars
```

`Artur_Ito.txt`

```
El origen
Star Wars
```

Desarrolle un programa que solicite como entrada el nombre de cierta película, y muestre por pantalla el total de dinero recaudado por esa película en este cine.

Además, debe generar un archivo de texto cuyo nombre coincida con el nombre COMPLETO de la película, separado por espacios en blanco (por ejemplo: `El origen.txt`), que tenga como contenido el nombre completo de todos los clientes que la han visto al menos una vez (el nombre no debe estar repetido y no importa el orden en que se escriban). Es importante considerar que si ningún cliente ha visto la película indicada, el archivo no debe crearse.

Nota: Los puntos suspensivos en los archivos reflejan que pueden existir muchos más datos.

Ejemplo

```
Star Wars
Dinero recaudado por Star Wars : $6000
```

`Star Wars.txt`

```
Luca Caminacielos
Artur Ito
```