UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

SEDE VIÑA DEL MAR

Programación

Unidad 9: Archivos

Ingeniería en Informática

Profesor: Gabriel Jara

gabriel.jara@usm.cl

Primer Semestre 2022

Para abrir un archivo

1. Crear una variable y asignarla al retorno de ***open('nombre_archivo.extensión')***
2. Recorrer con un ciclo ***for*** línea por línea.
3. Operar sobre cada línea (ej: print)
4. Cerrar el archivo con ***.close()***.

Para abrir un archivo

Cuando usa ***open()*** puede seleccionar un modo:

r	lectura
w	escritura
a	añadir

Para crear un archivo (o reemplazarlo)

1. Crear una variable y asignarla al retorno de ***open('nombre_archivo.extensión', 'w')***
2. Escriba cada línea del archivo con ***.write()***
3. Cerrar el archivo con ***.close()***.

Funciones sobre strings

Dado que al abrir un archivo y recorrer sus líneas vamos a obtener ***strings***, es útil disponer de funciones para manipular texto.

- ***replace()***
- ***split()***
- ***strip()***
- ***join()***

```
>>> 'Hola y chao'.replace('a','4')
'Hol4 y ch4o'
>>>
```

```
>>> 'Hola y chao'.split()
['Hola', 'y', 'chao']
>>> 'Hola y chao'.split(' y ')
['Hola', 'chao']
```

```
>>> ' Hola y chao '.strip()
'Hola y chao'
```

```
>>> print(' (X_X) '.join(['uno','dos','tres']))
...
uno (X_X) dos (X_X) tres
>>>
```

Caracteres especiales

Podemos imprimir un salto de línea con ***\n***

Podemos imprimir un tab con ***\t***

```
>>> print('Hola\tMundo\nOtro hola\t\tmundo\n')
Hola    Mundo
Otro hola               mundo

>>> print('Usando un \\ antes de un \\ se imprime un \\')
Usando un \ antes de un \ se imprime un \
>>>
```

format

Usando ***format()*** podemos ingresar intercalar variables en un texto.

```
>>> print('Abrimos con {}, cerramos con {}'.format('open','close'))
Abrimos con open, cerramos con close
>>>
```

Viajeros I 1/2

```
456:Cristina Palermo:Argentina:07-05-2019
523:Sophia Watson:EEUU:02-05-2019
364:Alan Moore:EEUU:12-12-2018
111:Karen Lee:EEUU:05-09-2019
646:Nicol´as Cabrera:Venezuela:01-01-2019
```

Tiene un archivo de viajeros de la forma que se presenta. Cree una función ***filtrar(nombre_archivo, año)*** recibe dos parámetros: un string con el nombre de un archivo como el anterior, y un entero que especifica un año. Esta función retorna un diccionario cuyas llaves son los países de procedencia de los viajeros en el año ingresado como parámetro. El valor asociado a cada llave es una lista de strings con los nombres de las personas de ese país que viajaron el año indicado.

```
>>> print(filtrar('inmigrantes.txt', 2019))
{'Argentina': ['Cristina Palermo'],   'EEUU': ['Sophia Watson',
'Karen Lee'], 'Venezuela': ['Nicolás Cabrera'    ]}
```

Viajeros I 2/2

```
456:Cristina Palermo:Argentina:07-05-2019
523:Sophia Watson:EEUU:02-05-2019
364:Alan Moore:EEUU:12-12-2018
111:Karen Lee:EEUU:05-09-2019
646:Nicol´as Cabrera:Venezuela:01-01-2019
```

Ordene el código para lograr la función solicitada.

```
nombre = campos[1]
dicc[pais].append(nombre)
if int(fecha[2]) == año:
dicc[pais] = []
archivo.close()
for linea in archivo:
pais = campos[2]
if pais not in dicc:
archivo = open(nombre_archivo, 'r')
dicc = {}
return dicc
fecha = campos[3].split('-')
def filtrar(nombre_archivo, año):
campos = linea.strip().split(':')
```

Viajeros II

```
456:Cristina Palermo:Argentina:07-05-2019
523:Sophia Watson:EEUU:02-05-2019
364:Alan Moore:EEUU:12-12-2018
111:Karen Lee:EEUU:05-09-2019
646:Nicol´as Cabrera:Venezuela:01-01-2019
```

escriba la función ***contar_ingresos(nombre_archivo, año)*** que recibe dos parámetros: un string con el nombre de un archivo con el formato indicado, y un entero que especifica un año. La función debe retornar un diccionario cuyas llaves sean los países de procedencia de los viajeros en el año señalado como parámetro. El valor de cada llave debe ser la cantidad de personas de ese país que viajó el año indicado.

```
>>> print(contar_ingresos('inmigrantes.txt', 2019))
{'Argentina': 1, 'EEUU': 2, 'Venezuela': 1}

>>> print(contar_ingresos('inmigrantes.txt', 2018))
{'EEUU': 1}
```

Viajeros III 1/2

```
456:Cristina Palermo:Argentina:07-05-2019
523:Sophia Watson:EEUU:02-05-2019
364:Alan Moore:EEUU:12-12-2018
111:Karen Lee:EEUU:05-09-2019
646:Nicol´as Cabrera:Venezuela:01-01-2019
```

Cree la función ***escribir_ingresos(archivo, año)*** que recibe dos parámetros: un string con el nombre de un archivo como el descrito, y un entero que especifica un año. La función debe crear un archivo de nombre ingresosX.txt, donde X corresponde al año ingresado como parámetro. El archivo debe contener los países de procedencia de los viajeros en el año especificado en el parámetro, ordenados de mayor a menor según cantidad de personas de ese país que viajaron ese año. Debe incluir en cada línea su número correlativo partiendo desde el 1. Además, la función debe retornar el número total de personas que ingresaron al país ese año.

Viajeros III 2/2

```
456:Cristina Palermo:Argentina:07-05-2019
523:Sophia Watson:EEUU:02-05-2019
364:Alan Moore:EEUU:12-12-2018
111:Karen Lee:EEUU:05-09-2019
646:Nicol´as Cabrera:Venezuela:01-01-2019
```

Ejemplo 1:

```
>>> print(escribir_ingresos('inmigrantes.txt', 2019))
4
```

ingresos2019.txt

```
1-EEUU:2
2-Venezuela:1
3-Argentina:1
```

Ejemplo 2:

```
>> escribir_ingresos('inmigrantes.txt', 2018)
1
```

ingresos2018.txt

```
1-EEUU:1
```

Perros I 1/2

Cesar Pyllan, el encantador de perros, mantiene el registro de las mascotas que ha atendido en un archivo cuya estructura es **`fecha;nombre;raza;problema;solucion`**. Un ejemplo de este archivo sería:

perros.txt

```
01-01-2015;boby;salchicha;muerde;collar de ahorque
02-03-2016;billy;pastor aleman;miedo;toque al costado
10-12-2016;boby;san bernardo;muerde;toque al costado
24-01-2017;willson;salchicha;miedo;toque al costado
04-02-2017;willson;san bernardo;miedo;toque al costado
03-02-2018;firulais;labrador;estres;SHHH
03-03-2018;firulais;pequines;estres;SHHH
...
```

Perros I 2/2

```
>>> print(leer_perros("perros.txt"))
{'boby': ['collar de ahorque', 'toque al costado'], 'billy': ['toque al
costado'],
'willson': ['toque al costado', 'toque al costado'], 'firulais': ['SHHH',
'SHHH']}
```

La función **leer_perros(na)** se comporta de la manera que se ve arriba.

El código de la función se encuentra desordenado. Usted debe ordenar las líneas e identarlas correctamente.

```
_,n,r,p,s = l.strip().split(';')
pe[n].append(s)
return pe
a = open(na)
a.close()
def leer_perros(na):
pe[n] = []
for l in a:
if n not in pe:
pe = {}
```

Perros II

En esta oportunidad César Pyllan le solicita su ayuda para lo siguiente:

Implemente la función **leer_razas(nombre_archivo)**, que recibe un string con el nombre de un archivo y clasifica los animales por raza. La función retorna un diccionario cuya llave es el nombre de la raza y el valor una lista con los nombres de los perros que pertenecen a ella. Guíese por el ejemplo a continuación.

```
>>> print(leer_razas('perros.txt'))
{'salchicha': ['boby', 'willson'], 'pastor aleman': ['billy'],
'san bernardo': ['boby', 'willson'], 'labrador': ['firulais'],
'pequines': ['firulais']}
```

Perros III

César ha notado que existen perros con el mismo nombre en distintas razas. Sabiendo que los nombres de los perros son únicos, esto sólo puede significar que existen mestizos. Desarrolle una función llamada **mestizos(raza1,raza2, nombre_archivo)** que recibe dos razas de perro y el nombre del archivo.

Esta función debe retornar una lista con los nombres de los perros mestizos de dichas razas. Si no existen perros mestizos para las razas consultadas, la función debe retornar una lista vacía. Guíese por los ejemplos a continuación.

```
>>> print(mestizos('salchicha','san bernardo','perros.txt'))
['boby', 'willson']
>>> print(mestizos('pequines','labrador','perros.txt'))
['firulais']
>>> print(mestizos('pequines','pastor aleman','perros.txt'))
[]
```

Línea Aérea

La línea aérea LANX desea premiar a sus pasajeros frecuentes incrementando sus kilómetros LANX-Pass de acuerdo al kilometraje acumulado a la fecha. para ello, cuenta con el archivo pasajeros.txt con campos separados por “:” con la siguiente estructura: rut:nombre:kilometraje, tal como se muestra en el siguiente ejemplo:

```
10043499-7:Javier Diaz:553450
7655511-7:Sebastian Rojas:134987
17273221-4:Elizabeth Bugueno:342021
18653834-4:Benjamin Bueno:2000
```

Considere que el archivo anterior puede tener muchos datos. Lo anterior es sólo un ejemplo

Línea Aérea I

Desarrolle la función **crear_lista(archivo)** que reciba como parámetro el nombre del archivo y retorne una lista de lista con los datos de cada pasajero. El campo kilometraje debe transformarse a entero.

```
>>>crear_lista("pasajeros.txt")}
[
['10043499-7', 'Javier Diaz', 553450], ['7655511-7', 'Sebastian Rojas', 134987],
['17273221-4', 'Elizabeth Bugueno', 342021] ['18653834-4', 'Benjamin Bueno', 2000]
]
```

Línea Aérea II

Desarrolle la función kilometraje(lista) que reciba como parámetro la lista creada anteriormente, e incremente en ella el kilometraje de cada pasajero de acuerdo a la siguiente tabla. La función retorna la lista actualizada.

KM Actual	KM a incrementar
0-15.000	3.000
15.001 - 50.000	6.000
50.001 - 150.000	10.000
más de 150.000	40.000

```
>>>kilometraje(lista)
[  ['10043499-7', 'Javier Diaz', 593450], ['7655511-7', 'Sebastian Rojas', 144987],
['17273221-4', 'Elizabeth Bugueno', 382021], ['18653834-4', 'Benjamin Bueno', 5000]]
```

Línea Aérea III

Desarrolle la función **escribir_archivo(archivo,lista)**, la cual recibe como parámetros el nombre de un archivo y la lista actualizada. La función debe guardar la información contenida en la lista, en un archivo con el mismo formato que el archivo original, es decir: rut:nombre:kilometraje.

```
>>>escribir_archivo("salida.txt", lista)
```

```
10043499-7:Javier Diaz:593450
7655511-7:Sebastian Rojas:144987
17273221-4:Elizabeth Bugueno:382021
18653834-4:Benjamin Bueno:5000
```

UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

SEDE VIÑA DEL MAR

Programación

Ingeniería en Informática

Profesor: Gabriel Jara

gabriel.jara@usm.cl

Primer Semestre 2022