

# TEMA 5

## Escribir y leer un archivo de texto



## ÍNDICE

|                                |    |
|--------------------------------|----|
| Escribir en un archivo         | 2  |
| Leer un archivo                | 13 |
| Instrucciones de código usadas | 20 |
| Reto 1                         | 21 |
| Reto 2                         | 21 |

## Escribir en un archivo

Todos los scripts que hemos hecho hasta ahora tienen un pequeño hándicap. Todo cambio en el programa que se realiza durante la ejecución del mismo (valores añadidos a listas o diccionarios, variables, etc) se pierden una vez cerramos el programa.

Hay muchas maneras de guardar estos cambios y de volver a utilizarlos una vez se vuelve a ejecutar el programa. Vamos a ver una forma de escribir en un archivo de texto y de leer posteriormente el contenido del archivo.

De la misma forma que imprimimos valores de nuestros scripts, podemos guardar en archivos información con cualquier extensión. El proceso de guardado y reutilización del archivo para futuras ejecuciones del script no es complejo, pero sí tiene muchas posibilidades así que vamos a empezar con algo asequible y tras ello se puede ir iterando y mejorando el uso de estas funciones de Python.

Para cualquier acción que requiera usar un archivo (ya sea escribir en el mismo o leer del mismo) vamos a tener que, en primera instancia, abrirlo. Si le indicamos a Python que queremos abrir un archivo, Python buscará el archivo indicado en la ubicación donde está el script que está ejecutando y, de no existir dicho archivo, creará uno con el nombre indicado.

Al abrirlo, así mismo, debemos indicar en qué modo lo abre, pues Python tiene varias posibilidades en esta línea:

- Abrirlo en modo lectura.
- Abrirlo en modo edición o escritura al final del mismo.
- Abrirlo en modo escritura desde cero, borrando lo que contiene.

Empecemos con el modo escritura, ya que es lo que inicialmente vamos a necesitar. Voy a reutilizar un código del tema pasado para tal fin:

```
nombres = []
cantidad_nombres = input ("¿Cuántos nombres quieres introducir en la
lista?: ")
cantidad_nombres = int (cantidad_nombres)
for numero in range(cantidad_nombres):
    numero_texto = str(numero + 1)
    nombre_elegido = input("Nombre número "+numero_texto+": ")
    nombres.append(nombre_elegido)
print (nombres)
```

Recuerda que dicho programa nos permitía almacenar nombres en una lista. Vamos ahora a hacer que el script, al finalizar, almacene los nombres en un archivo para poder tenerlos guardados. Para ello tendremos que hacer una secuencia de tres acciones muy sencillas:

1. Abrir el archivo en modo escritura.
2. Escribir la información deseada en el archivo.
3. Cerrar el archivo.

Es importante siempre cerrar un archivo tras haber terminado de usarlo o podemos tener problemas de acceso a la información que contenga si no lo hacemos.

Veamos cómo sería el código para guardar la lista nombres:

```
nombres = []
cantidad_nombres = input ("¿Cuántos nombres quieres introducir en la
lista?: ")
cantidad_nombres = int (cantidad_nombres)
for numero in range(cantidad_nombres):
    numero_texto = str(numero + 1)
    nombre_elegido = input("Nombre número "+numero_texto+": ")
    nombres.append(nombre_elegido)
print (nombres)
archivo = open("nombres.txt","w")
for nombre in nombres:
    archivo.write (nombre + "\n")
archivo.close()
```

Como ves, es sencillo. Debemos indicar a Python que abra un archivo y guardamos el archivo abierto bajo un nombre, en nuestro caso *archivo* (a efectos prácticos podemos decir que está utilizando una variable para ubicar el archivo abierto). Posteriormente indicamos que queremos que el archivo se abra en modo "w", es decir, modo escritura (*write* en inglés).

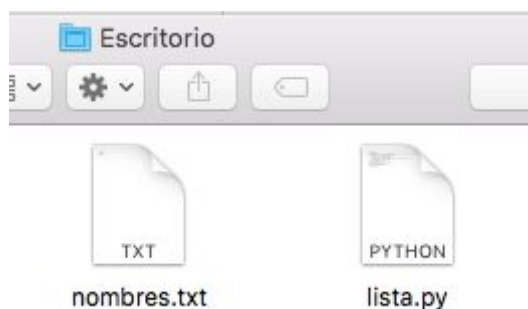
Tras abrir el archivo indicamos a Python lo que queremos almacenar en el archivo con la función *.write*. En nuestro caso recorremos la variable *nombres* con un *for* y vamos guardando cada elemento que contiene en el archivo. El comando *\n* indica un salto de carro, es decir, que separe los elementos en líneas diferentes.

Tras ello indicamos que queremos cerrar el archivo con la función *.close()*

Veamos su ejecución guardando tres nombres:

```
[MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$ python3 lista.py
¿Cuántos nombres quieres introducir en la lista?: 3
Nombre número 1: Alfredo
Nombre número 2: María
Nombre número 3: Pedro
['Alfredo', 'María', 'Pedro']
MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$
```

Hasta aquí no hay ninguna diferencia aparente con el código del tema anterior, pero veamos qué ha pasado en mi escritorio:



Se ha generado un archivo `.txt` llamado `nombres`. Dicho archivo debería contener los nombres introducidos en la lista del script, veamos que contiene abriéndolo:

A screenshot of a text editor window titled 'nombres.txt — Modificado'. The window contains the following text:

```
Alfredo
María
Pedro
|
```

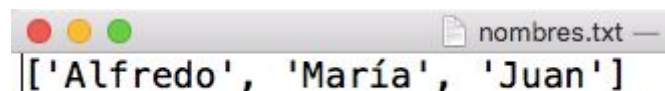
Efectivamente, se han guardado los nombres que pusimos en la lista. Prueba todo lo anterior en tu equipo antes de seguir.

Python no permite almacenar una lista directamente, sólo permite almacenar strings, por ello hemos recorrido la lista para almacenar sus elementos, que eran strings. Pero si tenemos una lista que contiene otro tipo de datos o bien queremos guardar la lista entera debemos convertirla en string previamente. Veamos cómo sería:

```
nombres = []
cantidad_nombres = input ("¿Cuántos nombres quieres introducir en la
lista?: ")
```

```
cantidad_nombres = int (cantidad_nombres)
for numero in range(cantidad_nombres):
    numero_texto = str(numero + 1)
    nombre_elegido = input("Nombre número "+numero_texto+": ")
    nombres.append(nombre_elegido)
print (nombres)
archivo = open("nombres.txt", "w")
nombres = str(nombres)
archivo.write(nombres)
archivo.close()
```

El único cambio consiste en convertir la lista *nombres* a string y posteriormente la imprimimos en el archivo. Una vez guardado y ejecutado vuelve a crear un archivo, pero fijaros en la estructura que tiene ahora la información que contiene:



```
['Alfredo', 'María', 'Juan']
```

Como puedes ver, al convertir a string una lista, Python mantiene todas las partes de su estructura (los corchetes, las comillas, las comas...). Por lo que puede ser una forma de almacenar los datos algo más incómoda para trabajar posteriormente con ellos.

Nos quedamos con la opción de escritura de elementos por líneas y vamos a ver cómo podríamos seguir escribiendo más nombres en un archivo ya creado. La opción para escribir nuevos nombres requiere que aprendamos a abrir archivos en modo escritura tras el contenido del archivo. Ese modo se denomina como "a" en vez de "w".

Si queremos añadir texto deberíamos dar al usuario la opción de decidir si quiere seguir añadiendo nombres o empezar de nuevo la lista. Observa este código a ver si entiendes todas sus partes:

```
nombres = []
intencion = input ("¿Quieres introducir más nombres en la lista
existente o quieres \n empezar una nueva lista? (1: Más nombres, 2:
Nueva lista): ")
cantidad_nombres = input("¿Cuántos nombres quieres introducir en la
lista?: ")
cantidad_nombres = int(cantidad_nombres)
for numero in range(cantidad_nombres):
    numero_texto = str(numero+1)
    nombre_elegido = input("Nombre número "+numero_texto+": ")
    nombres.append(nombre_elegido)
```

```
print(nombres)
if intencion == "1":
    archivo = open("nombres.txt","a")
    for nombre in nombres:
        archivo.write (nombre)
    archivo.close()
elif intencion == "2":
    archivo = open("nombres.txt","w")
    for nombre in nombres:
        archivo.write (nombre)
    archivo.close()
```

Simplemente preguntamos inicialmente si queremos escribir en un archivo nuevo (da igual que exista, el comando "w" sobrescribe todo el contenido) o bien escribir al final de un archivo existente. Tras elegir que opción queremos hacemos una cosa u otra en función de la respuesta.

Prueba el script anterior las veces que quieras, aquí tienes una captura de la prueba del mismo para crear una nueva lista primero y luego guardar nuevos nombres sobre la misma:

```
[MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$ python3 lista.py
¿Quieres introducir más nombres en la lista existente o quieres
empezar una nueva lista? (1: Más nombres, 2: Nueva lista): 2
¿Cuántos nombres quieres introducir en la lista?: 3
Nombre número 1: Alfredo
Nombre número 2: María
Nombre número 3: Jorge
['Alfredo', 'María', 'Jorge']
[MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$ python3 lista.py
¿Quieres introducir más nombres en la lista existente o quieres
empezar una nueva lista? (1: Más nombres, 2: Nueva lista): 1
¿Cuántos nombres quieres introducir en la lista?: 2
Nombre número 1: Rodrigo
Nombre número 2: Ana
['Rodrigo', 'Ana']
MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$ █
```

Y aquí el contenido del archivo de texto creado:

```
nombres.txt
Alfredo
María
Jorge
Rodrigo
Ana
```

¿Ya lo has probado? ¡Perfecto! Con las nociones que posees del tema anterior puedes recorrer cualquier lista o diccionario y guardarlo en un archivo en cualquiera de las dos formas que hemos visto. También puedes guardar el contenido de cualquier variable, e incluso podrías terminar siempre un programa preguntando con un `input` "¿Hay algo que quieras guardar?" y guardar aquello que el usuario escriba.

Es posible que te preguntes para qué iba alguien querer hacer algo tan enrevesado para reproducir una función directa de un archivo de texto, es decir, en un archivo de texto puedes guardar los nombres que quieras directamente... Bueno, la gracia viene cuando introduces las opciones de Python para que lo que almacenes tenga una verdadera utilidad.

Pongamos un ejemplo: quiero asignar un número a cada uno de los miembros de un grupo de personas. Podría simplemente introducirlas aleatoriamente en una lista y el número que le corresponde es el índice que ocupan en la lista (o bien el índice + 1 por aquello de que nadie tenga el número 0). Tras realizar la lista aleatoria se guarda en un archivo y así la tenemos para cuando queramos mirarla.

Veamos cómo sería ese programa para una lista de 5 personas. Lo primero que tenemos que aprender es a generar números aleatorios y para ello importaremos una nueva librería: `random`:

```
import random
```

A continuación vamos a generar una lista de tantos números aleatorios como necesitemos. Para ello podemos preguntar al usuario cuantos nombres quiere introducir aleatoriamente en una lista y utilizar esa información:

```
import random
cantidad = input("¿Cuántos nombres quieres introducir aleatoriamente en la lista?: ")
cantidad = int(cantidad)
```



Ya sabes cómo conseguir que el script pregunte una y otra vez al usuario si, al intentarlo, la respuesta dada no puede ser convertida en *int* por no ser un número entero.

Ahora vamos con el código para generar números aleatorios que no se repitan. Vamos a describirlo con lenguaje humano y posteriormente vemos cómo se dice en lenguaje Python:

*Genera un número aleatorio entre 1 y la cantidad de nombres que tendrá la lista. A continuación genera un nuevo aleatorio con la misma instrucción que no sea el primero. Posteriormente genera un tercero que no sea ninguno de los dos ya creados y así hasta crear tantos números aleatorios como nombres vaya a haber en la lista.*

Para Python esto se dice de una forma un poco diferente, pero lo único nuevo sería saber cómo se genera un número aleatorio. La instrucción para generar aleatorios es *random.randint* seguida de dos parámetros que indican entre qué dos valores debe estar el aleatorio: *random.randint(a,b)*, siendo a y b dichos valores, ambos incluidos.

Sigamos pues con nuestro script. Como tenemos que generar tantos números como personas queramos en la lista usaremos un *for*:

```
import random
cantidad = input("¿Cuántos nombres quieres introducir aleatoriamente en la lista?: ")
cantidad = int(cantidad)
for numero in range(cantidad):
    aleatorio = random.randint(1,cantidad)
    print(aleatorio)
```

Prueba el script anterior para que veas su funcionamiento. Como ya dijimos anteriormente, ten cuidado de no tener ningún archivo llamado random en el directorio donde se encuentre tu script. Prueba introduciendo un número en torno a 10.

Aquí dejo la captura de pantalla de la ejecución del programa para 10 números:

```
[MacBook-Air-de-Alfredo-2:Desktop_alfredosanchezsanchez$ python3 aleatorio.py
¿Cuántos nombres quieres introducir aleatoriamente en la lista?: 10
7 5 aleatorio = random.randint(1,cantidad)
6 aleatorio = random.randint(1,cantidad)
8 aleatorio = random.randint(1,cantidad)
5 aleatorio = random.randint(1,cantidad)
7 aleatorio = random.randint(1,cantidad)
9 aleatorio = random.randint(1,cantidad)
9 aleatorio = random.randint(1,cantidad)
1 aleatorio = random.randint(1,cantidad)
3 aleatorio = random.randint(1,cantidad)
6 aleatorio = random.randint(1,cantidad)
```

Como verás ha elegido 10 números aleatoriamente, pero algunos se repiten. Para que no se repita **podemos ir metiéndolos en una lista** y, al generar uno nuevo, comprobamos si está en la lista y si es así generamos uno nuevo, en caso contrario lo introducimos en la lista:

```
import random
cantidad = input("¿Cuántos nombres quieres introducir aleatoriamente en
la lista?: ")
cantidad = int(cantidad)
aleatorios = []
for numero in range(cantidad):
    aleatorio = random.randint(1,cantidad)
    while aleatorio in aleatorios:
        aleatorio = random.randint(1,cantidad)
    aleatorios.append(aleatorio)
    print(aleatorio)
```

Como puedes ver, simplemente creamos un aleatorio y comprobamos si está en la lista *aleatorios* y mientras así sea seguimos generando números aleatorios. Una vez no se cumpla esta premisa pasaremos a almacenar el número en la lista *aleatorios* e imprimirlo.

Pruébalo, aquí tienes la captura que certifica su funcionamiento:

```
[MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$ python3 aleatorio.py
¿Cuántos nombres quieres introducir aleatoriamente en la lista?: 10
6
3
10
2
5
8
7
9
1
4
```

Volvamos a nuestro programa. Ahora que tenemos esa lista de aleatorios sólo tenemos que generar la otra lista, la de nombres que hay que organizar aleatoriamente.

```
import random
cantidad = input("¿Cuántos nombres quieres introducir aleatoriamente en
la lista?: ")
cantidad = int(cantidad)
aleatorios = []
for numero in range(cantidad):
    aleatorio = random.randint(1,cantidad)
    while aleatorio in aleatorios:
        aleatorio = random.randint(1,cantidad)
    aleatorios.append(aleatorio)
nombres = []
for i in range (cantidad):
    nombreNumero = str(i+1)
    nombre = input("Nombre "+nombreNumero+": ")
    nombres.append(nombre)
print(aleatorios)
print(nombres)
```

Intenta entender el código anterior y pruébalo hasta que tengas claro cómo está funcionando.

Aquí tienes una captura de su ejecución:

```
[MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$ python3 aleatorio.py
¿Cuántos nombres quieres introducir aleatoriamente en la lista?: 5
Nombre 1: Alfredo
Nombre 2: Jorge
Nombre 3: Ana
Nombre 4: Rodrigo
Nombre 5: María
[5, 2, 4, 1, 3]
['Alfredo', 'Jorge', 'Ana', 'Rodrigo', 'María']
```

Si observas la captura puedes ver que tenemos dos listas, una con números y otra con nombres. Ahora lo más sencillo sería vincular ambas listas de manera que el índice 0 de una lista se asigne al índice 0 de la otra. Podemos hacer un diccionario en el cual la clave sea el nombre y el valor sea el número:

```
import random
cantidad = input("¿Cuántos nombres quieres introducir aleatoriamente en
la lista?: ")
cantidad = int(cantidad)
aleatorios = []
for numero in range(cantidad):
    aleatorio = random.randint(1,cantidad)
    while aleatorio in aleatorios:
        aleatorio = random.randint(1,cantidad)
    aleatorios.append(aleatorio)
nombres = []
for i in range(cantidad):
    nombreNumero = str(i+1)
    nombre = input("Nombre "+nombreNumero+": ")
    nombres.append(nombre)
print(aleatorios)
print(nombres)
diccionario = {}
for i in range(len(aleatorios)):
    diccionario[nombres[i]] = aleatorios[i]
print(diccionario)
```

Una vez más, prueba el código guardándolo como script y asegúrate de entender cómo funciona. Es importante tener claro esto antes de pasar al siguiente punto. Dejo una vez más una captura de su ejecución para 4 nombres:

```
[MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$ python3 aleatorio.py
¿Cuántos nombres quieres introducir aleatoriamente en la lista?: 4
Nombre 1: Ana
Nombre 2: Mario
Nombre 3: Pedro
Nombre 4: Yolanda
[3, 2, 1, 4]
['Ana', 'Mario', 'Pedro', 'Yolanda']
{'Yolanda': 4, 'Ana': 3, 'Pedro': 1, 'Mario': 2}
```

Si ya has comprendido el código anterior llega el momento de que intentes guardar en un archivo el diccionario que hemos generado con el script. Recuerda guardarlo como string antes. Ahora si empieza a tener algo de sentido este script, ¿verdad? Guardamos un diccionario con números aleatorios y lo tenemos para cuando queramos revisar qué número tenía asignado cada nombre.

## Leer un archivo

Vamos ahora a ver cómo podemos sacar información de un archivo. El proceso es sencillo, simplemente tenemos que indicar a Python que queremos abrir el archivo en modo lectura ("r") e indicar que queremos leerlo. Python automáticamente guardará el contenido del archivo como un string. Veamos cómo abrir el archivo creado anteriormente para guardar nombres:

```
archivo = open("nombres.txt", "r")
contenido = archivo.read()
print (contenido)
archivo.close()
```

Guarda el anterior código como script y ejecútalo. Asegúrate de que exista el archivo "nombres.txt" en la ruta donde guardes el script. Si no fuese así crea uno con una lista de nombres (con un nombre en cada línea).

Si tratamos de abrir un archivo que no existe Python nos dará error. Al guardar en archivo sí crea uno en caso de no existir, pero esto no ocurre a la hora de abrir un archivo no existente:

```
>>> archivo = open("hola_mundo.txt", "r")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
FileNotFoundError: [Errno 2] No such file or directory: 'hola_mundo.txt'
>>>
```

Guarda el anterior código como script y ejecútalo. Ase

También se puede imprimir directamente el contenido del archivo, sin guardarlo:

```
archivo = open("nombres.txt", "r")
print(archivo.read())
archivo.close()
```

Cuidado al abrir archivos que contengan tildes o símbolos no anglosajones desde el IDLE, puede ocurrir que se produzcan errores. En general, como indicamos en el primer tema, no es buena práctica usar cualquier tipo de caracteres que no sean anglosajones (tildes, interrogaciones y admiraciones iniciales, etc).

Otra forma de abrir un archivo es leerlo línea a línea, el código sería el siguiente:

```
archivo = open("nombres.txt", "r")
```

```
for linea in archivo:  
    print(linea)  
archivo.close()
```

Prueba el código para que veas cómo hace lo mismo que el anterior, si bien el espacio entre las líneas al imprimirlas es doble.

Podemos utilizar el código anterior para ir almacenando el contenido de cada línea en una lista:

```
archivo = open("nombres.txt", "r")  
nombres = []  
for linea in archivo:  
    nombres.append(linea)  
archivo.close()  
print(nombres)
```

Parece que el anterior código debería funcionar sin problemas. Pruébalo como script antes de seguir.

El código anterior está funcionando correctamente, pero tiene un pequeño problema...

```
MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$ python3 abrir_archivo.py  
['Alfredo\n', 'Maria\n', 'Jorge\n', 'Rodrigo\n', 'Ana\n']  
MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$
```

... al final de cada línea del archivo de texto hay un salto de carro y como Python es muy obediente lo ha incluido como parte de cada nombre con lo cual la lista *nombres* tiene que ser depurada. Esto es algo habitual al incorporar caracteres de texto de un archivo a un programa, generalmente tendremos que eliminar algunos caracteres o bien sustituirlos por otros.

Veamos cómo podemos eliminar ese incómodo salto de carro (`\n`):

```
archivo = open("nombres.txt", "r")  
nombres = []  
for linea in archivo:  
    linea = linea.replace("\n", "")  
    nombres.append(linea)  
archivo.close()  
print(nombres)
```

La función *replace* nos permite, en un string, sustituir unos caracteres por otros. En mi programa estoy sustituyendo el salto de carro por nada (poner dos dobles comillas indica que quieres poner lo que contienen las dobles comillas en un string, y como no contienen nada, no pondremos nada).

Prueba el código anterior, si no lo entiendes bien pruébalo incluyendo algún carácter entre las segundas dobles comillas, por ejemplo un signo de admiración:

```
archivo = open("nombres.txt","r")
nombres = []
for linea in archivo:
    linea = linea.replace("\n","!")
    nombres.append(linea)
archivo.close()
print(nombres)
```

Si vemos la ejecución del programa anterior nos daremos cuenta que ha sustituido el salto de carro por una admiración:

```
MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$ python3 abrir_archivo.py
['Alfredo!', 'María!', 'Jorge!', 'Rodrigo!', 'Ana!']
MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$
```

Perfecto, ya hemos conseguido leer un archivo. Ahora vamos a ver otro problema que podemos encontrarlos.

¿Qué ocurre si hemos guardado una lista como tal directamente en el archivo? Si recordáis, al principio del tema veíamos este caso:



```
nombres.txt
['Alfredo', 'María', 'Juan']
```

Si incorporamos esa lista al programa tendremos que ver cómo sustituimos más cosas, no sólo el salto de carro. Vamos a crear el archivo de texto que contenga esa información y vemos cómo podemos ir trabajando ese string para convertirlo en una lista.

Crea un archivo *nombres.txt* con una lista de tres nombres como la de arriba y ábrela con el script creado anteriormente antes de seguir. No pongas tilde en María en caso de estar usando la IDLE.



Aquí tenéis la captura de lo que está ocurriendo ahora:

```
MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$ python3 abrir_archivo.py
[["Alfredo", "Maria", "Juan"]] Atajo de teclado:
MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$
```

Menudo follón. Tenemos una lista que contiene un único elemento lleno de comas, comillas, etc. Además, las comillas se han convertido en comillas iniciales y finales... Esto nos va a llevar un rato.

Empecemos eliminando varias cosas que no necesitamos:

```
archivo = open("nombres.txt", "r")
nombres = []
for linea in archivo:
    linea = linea.replace("\n", "")
    linea = linea.replace("[", "")
    linea = linea.replace("]", "")
    linea = linea.replace("'", "")
    linea = linea.replace('"', "")
    nombres.append(linea)
archivo.close()
print(nombres)
```

Puede parecer confuso porque este editor de texto interpreta los caracteres, a ver si poniendo captura de un editor de código lo tenéis más claro:

```
archivo = open("nombres.txt" , "r")
nombres = [ ]
for linea in archivo:
    """Sustituimos salto de carro"""
    linea = linea.replace("\n","")
    """Sustituimos corchetes"""
    linea = linea.replace("[","")
    linea = linea.replace("]","")
    """Sustituimos comillas iniciales"""
    linea = linea.replace("'", "")
    """Sustituimos comillas finales"""
    linea = linea.replace('"', "")
    nombres.append(linea)
archivo.close ( )
print(nombres)
```

Prueba a guardar el código como un script y ejecutarlo para ver qué ocurre. Cuidado con las comillas, es complicado poner unas comillas en un editor de código, lo más fácil a veces es copiarlas de algún sitio y pegarlas porque el editor, al escribirlas con el teclado, las pone como comillas rectas.

¿Ya lo has probado? El resultado deberías ser el siguiente:

```
MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$ python3 abrir_archivo.py
['Alfredo,Maria,Juan']
MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$
```

Parece que ya tenemos una lista al uso, pero hay que tener cuidado porque las apariencias engañan. Si te fijas, los nombres están todos englobados bajo unas únicas comillas simples, esto quiere decir que todo ello es un único elemento de la lista. Las listas guardan cada elemento (en caso de ser un string) englobado en comillas, nuestra lista debería tener un aspecto similar a este:

```
["Alfredo","Maria","Juan"]
```

Para ello vamos a usar un último elemento que parte los strings por un caracter concreto que tendremos que indicar, convirtiéndolos en listas con tantos elementos como los que queden al partir el string:

```
archivo = open("nombres.txt","r")
```

```
nombres = []
for linea in archivo:
    linea = linea.replace("\n", "")
    linea = linea.replace("[", "")
    linea = linea.replace("]", "")
    linea = linea.replace("'", "")
    linea = linea.replace('"', "")
    linea = linea.split(",")
    nombres.append(linea)
archivo.close()
print(nombres)
```

Prueba el código anterior, debería salirte algo similar a esto:

```
MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$ python3 abrir_archivo.py
['Alfredo', 'Maria', 'Juan']
MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$
```

El comando `.split` parte un string por el caracter indicado, eliminándolo. Si indicamos a Python que parta por las comas, Python crea una lista de tantos elementos como comas haya mas uno, ya que la primera coma que sustituimos nos dará una lista de 2 elementos y posteriormente cada nueva coma añade un nuevo elemento.

Por lo tanto pasamos a tener una lista dentro de nuestra lista. Para evitar este efecto sólo tendríamos que ir recorriendo la lista generada e introduciendo sus elementos en nuestra lista `nombres`:

```
archivo = open("nombres.txt", "r")
nombres = []
for linea in archivo:
    linea = linea.replace("\n", "")
    linea = linea.replace("[", "")
    linea = linea.replace("]", "")
    linea = linea.replace("'", "")
    linea = linea.replace('"', "")
    linea = linea.split(",")
    for elemento in linea:
        nombres.append(elemento)
archivo.close()
print(nombres)
```

¡Hecho! guárdalo en un script y ejecútalo intentando entender todo lo que ha ocurrido antes de continuar.

Aquí tienes una captura de cómo nuestro archivo ha sido, por fin, incorporado como lista a un script de Python:

```
MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$ python3 abrir_archivo.py
['Alfredo', 'Maria', 'Juan']
MacBook-Air-de-Alfredo-2:Desktop alfredosanchezsanchez$
```

Ahora toca trabajar para afianzar todo lo visto en este tema.

## Instrucciones de código usadas

```
#Abrir un archivo en modo escritura, borrando el contenido
archivo = open("nombre.txt","w")
#Escribir contenido en un archivo
archivo.write(contenido)
#Cerrar un archivo
archivo.close()
#Abrir un archivo en modo escritura tras el contenido existente
archivo = open("nombre.txt","a")
#Importar biblioteca para generar aleatorios
import random
#Crear un número aleatorio entre dos valores y guardarlo en una variable
variable = random.randint(a,b)
#Abrir un archivo en modo lectura
archivo = open("nombre.txt","r")
#Leer el contenido de un archivo
archivo.read()
#Sustituir caracteres de un string
string.replace("caracter_antiguo","caracter_nuevo")
#Partir un string por un caracter concreto
string.split("caracter")
```

## Reto 1

Trata de crear un script que guarde una lista de nombres en un archivo ya existente que contenga una lista, leyendo previamente la lista. Inicialmente tendremos que abrir el archivo con cuidado de que, en caso de no existir, Python nos dará un error (prueba con un `try - except`).

El script nos tendrá que dar la opción de utilizar los nombres existentes en la lista y seguir incorporando nuevos o bien eliminarlos todos y empezar de cero.

## Reto 2

Crea un script que te permita ir almacenando en un archivo una lista de la compra. Utiliza un diccionario para almacenar cada artículo que hay que adquirir y la cantidad necesaria (tendrás que ir preguntando al usuario y generar dos listas, una de artículos y otra de cantidades). El usuario debería poder elegir entre introducir un nuevo artículo o terminar la lista. Una vez terminada la lista habrá que guardarla en un archivo de texto en un formato legible (un artículo por línea, por ejemplo).