

1 DE DICIEMBRE DE 2025

# EJERCICIO CON ARCHIVOS

ENRIQUE GONZÁLEZ



CAMPUSFP  
1º DAM GETAFE

## INDICE

Ejercicio 1 (archivo de texto).....	2
Ejercicio 2 (archivo de texto).....	4
Ejercicio 3 (serialización con pickle) .....	6
Ejercicio 4 (serialización con pickle) .....	8

## Ejercicio 1 (archivo de texto)

### Guardar una lista de nombres en un archivo

Escribe un programa en Python que haga lo siguiente:

1. Pida al usuario que introduzca una serie de nombres uno a uno.
2. La introducción de nombres terminará cuando el usuario escriba la palabra FIN.
3. Todos los nombres introducidos deben almacenarse en una lista.
4. Finalmente, el programa deberá crear un archivo de texto llamado nombres.txt y guardar cada nombre en una línea distinta.

Indicaciones para el alumno:

- Solo se admite texto (nombres).
- Si el archivo ya existe, debe sobrescribirse.
- El archivo debe quedar en la misma carpeta que el programa.

CÓDIGO:

```
nombres=[]

#METER NOMBRES EN LISTAS

nombre=str(input("Escribe un nombre de persona o 'fin' para acabar: "))

while nombre != "fin":

    nombres.append(nombre)

    nombre=str(input("Escribe otro nombre de persona o 'fin' para acabar: "))

#CREAR Y ESCRIBIR ARCHIVO

with open("nombres.txt", "w") as archivo:

    for nombre in nombres:

        archivo.write(nombre + "\n") #Mostrar de uno en uno

print("Archivo 'nombres.txt' creado.")
```

ENRIQUE GONZÁLEZ  
1º DAM GETAFE

#MOSTRAR ARCHIVO CREADO(DA ERROR SI NO SE HA HECHO EL PASO ANTERIOR)

with open("nombres.txt", "r") as archivo:

for linea in archivo:

print(linea.strip())

```
PS C:\Users\CampusFP\AppData\Local\Programs\Microsoft VS Code> & C:\U
nombres=[].py"
Escribe un nombre de persona o 'fin' para acabar: kike
Escribe otro nombre de persona o 'fin' para acabar: arturo
Escribe otro nombre de persona o 'fin' para acabar: fin
Archivo 'nombres.txt' creado.
kike
arturo
PS C:\Users\CampusFP\AppData\Local\Programs\Microsoft VS Code> r
```

## Ejercicio 2 (archivo de texto)

### Registro de temperaturas diarias

Desarrolla un programa que cumpla los siguientes pasos:

1. El programa debe pedir al usuario que introduzca la temperatura media de cada día de la semana (lunes a domingo).
2. Cada temperatura debe ser un número decimal.
3. El programa debe guardar todas las temperaturas en un archivo llamado temperaturas.txt.
4. El archivo debe tener este formato exacto:

Lunes: X

Martes: X

Miércoles: X

...

5. Cuando acabe de guardar los datos, el programa debe mostrar un mensaje indicando que el archivo se ha generado correctamente.

CÓDIGO:

```
dias=["Lunes", "Martes", "Miércoles", "Jueves", "Viernes", "Sábado", "Domingo"]
```

```
temperaturas={}
```

```
#BUCLE TEMPERATURAS
```

```
for dia in dias:
```

```
    temperatura=float(input(f"Introduce la temperatura media del {dia}: "))
```

```
    temperaturas[dia]=temperatura
```

```
#GUARDAR
```

```
with open("temperaturas.txt", "w") as archivo:
```

```
    for dia in dias:
```

```
        archivo.write(f"{dia}: {temperaturas[dia]}\n")
```

```
print("Archivo 'temperaturas.txt' creado.")
```

```
#ENSEÑAR
```

with open("temperaturas.txt", "r") as archivo:

for linea in archivo:

print(linea.strip())

```
PS C:\Users\CampusFP\AppData\Local\Programs\Microsoft VS  
ombres=[].py"  
Introduce la temperatura media del Lunes: 6  
Introduce la temperatura media del Martes: 3.4  
Introduce la temperatura media del Miércoles: 2  
Introduce la temperatura media del Jueves: 6.7  
Introduce la temperatura media del Viernes: 7.7  
Introduce la temperatura media del Sábado: 4  
Introduce la temperatura media del Domingo: 2  
Archivo 'temperaturas.txt' creado.  
Lunes: 6.0  
Martes: 3.4  
Miércoles: 2.0  
Jueves: 6.7  
Viernes: 7.7  
Sábado: 4.0  
Domingo: 2.0
```

## Ejercicio 3 (serialización con pickle)

### Guardar y cargar un diccionario de alumnos

Escribe un programa que gestione un pequeño diccionario de alumnos utilizando la librería pickle.

1. El programa debe pedir al usuario que introduzca pares “nombre – nota”.
2. Estos datos deben guardarse en un diccionario con la estructura:

```
{"Ana": 8.5, "Luis": 6.0, ...}
```

3. Cuando el usuario escriba FIN, el programa debe guardar el diccionario completo en un archivo llamado alumnos.pkl utilizando pickle.
4. En una segunda parte del programa, o en una segunda ejecución, el programa debe abrir el archivo alumnos.pkl y mostrar por pantalla:
  - La lista completa de alumnos
  - La media de las notas

CÓDIGO:

```
import pickle

alumnos={}

nombre=""

#BUCLE NOMBRES/NOTA

print("Escribe alumnos y sus notas o 'fin' para salir.")

while nombre != "fin":

    nombre=input("Nombre del alumno: ")

    if nombre != "fin":

        nota=float(input("Nota del alumno: "))

        alumnos[nombre]=nota

#GUARDAR

with open("alumnos.pkl", "wb") as archivo:
```

ENRIQUE GONZÁLEZ  
1º DAM GETAFE

```
pickle.dump(alumnos, archivo)  
  
print("Diccionario guardado en 'alumnos.pkl'")
```

#MOVIDAS CON PICKLE

#CARGAR DICCIONARIO

```
with open("alumnos.pkl", "rb") as archivo:
```

```
    alumnos=pickle.load(archivo)
```

#SHOW LISTA ALUMOS

```
print("Lista de alumnos y sus notas:")
```

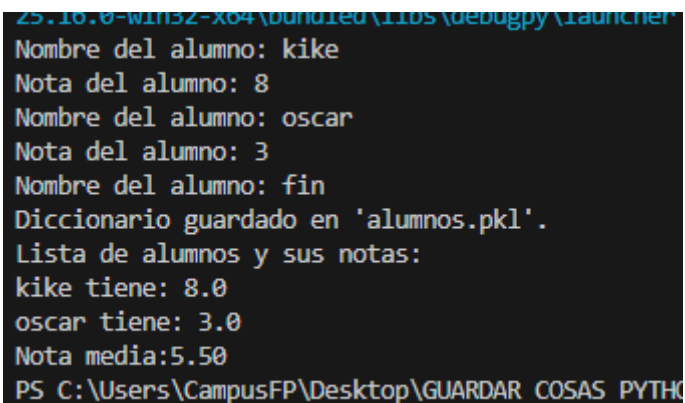
```
for nombre, nota in alumnos.items():
```

```
    print(f"{nombre} tiene: {nota}")
```

#NOTA MEDIA

```
media=sum(alumnos.values())/len(alumnos)
```

```
print(f"Nota media:{media:.2f}")
```



```
25.10.0-win32-x64 (bundled) (110s (debugpy) (launcher)  
Nombre del alumno: kike  
Nota del alumno: 8  
Nombre del alumno: oscar  
Nota del alumno: 3  
Nombre del alumno: fin  
Diccionario guardado en 'alumnos.pkl'.  
Lista de alumnos y sus notas:  
kike tiene: 8.0  
oscar tiene: 3.0  
Nota media:5.50  
PS C:\Users\CampusFP\Desktop\GUARDAR COSAS PYTHON>
```



## Ejercicio 4 (serialización con pickle)

### Sistema simple de inventario

Desarrolla un pequeño sistema de inventario utilizando clases y serialización.

1. Define una clase llamada Producto con los atributos:
  - nombre
  - precio
  - cantidad
2. El programa debe permitir:
  - a) Crear un nuevo producto.
  - b) Añadirlo a una lista de inventario.
  - c) Guardar la lista completa en un archivo inventario.pkl utilizando pickle.
3. En una segunda parte del programa, o en otra ejecución, deberás:
  - a) Cargar el archivo inventario.pkl.
  - b) Mostrar todos los productos con su formato:  
"Producto: nombre – Precio: X – Cantidad: X"

Si el archivo no existe, el programa debe informar al usuario y permitir crear un inventario nuevo.

CÓDIGO:

```
# EJERCICIO 4
```

```
# Importar pickle y os
```

```
import pickle
```

```
import os
```

```
# Clase Producto
```

```
class Producto:
```

```
    def __init__(self, nombre, precio, cantidad):
```

```
        self.nombre = nombre
```

```
        self.precio = precio
```

```
        self.cantidad = cantidad
```

```
    def __str__(self):
```

```
        return f"{self.nombre} - Precio: {self.precio}€ - Cantidad: {self.cantidad}"

# Función para cargar el archivo si existe
def cargar_inventario():
    if os.path.exists("inventario.pkl"):
        archivo = open("inventario.pkl", "rb")
        inventario = pickle.load(archivo)
        archivo.close()

        # Mostrar los productos que hay guardados
        for producto in inventario:
            print(f"Producto: {producto.nombre} - Precio: {producto.precio} - Cantidad: {producto.cantidad}")

        return inventario
    else:
        return []

# Función para crear producto y añadirlo a la lista
def crear_producto(inventario):
    # Preguntar nombre, precio y cantidad
    nombre = input("Nombre: ")
    precio = float(input("Precio: "))
    while precio < 0:
        print("Precio no válido, prueba otra vez")
        precio = float(input("Precio: "))
    cantidad = int(input("Cantidad: "))
    while cantidad < 0:
        print("Cantidad no válida, prueba otra vez")
        cantidad = int(input("Cantidad: "))
```

```
# Añadir el producto a la lista

p = Producto(nombre, precio, cantidad)

inventario.append(p)

print("Producto añadido")
```

```
# Función para guardar el inventario

def guardar_inventario(inventario):

    with open("inventario.pkl", "wb") as archivo:

        pickle.dump(inventario, archivo)
```

```
# Función para mostrar el menú

def mostrar_menu():

    print("--- MENÚ ---")

    print("1. Crear producto y añadirlo a la lista")

    print("2. Guardar el inventario")

    print("3. Salir")
```

```
#Función para el menú

def menu():

    #Cargar la lista del inventario o dejarla vacía

    inventario = cargar_inventario()

    mostrar_menu()

    opcion = int(input("Opción: "))

    while opcion != 3:

        match opcion:

            case 1:

                crear_producto(inventario)

            case 2:
```

```
        guardar_inventario(inventario)

    case 3:

        print("Saliendo...")

    case _:

        print("Opción no válida")

    # Mostrar menú nuevamente

    mostrar_menu()

    opcion = int(input("Opción: "))

#ARRANQUE

menu()
```

COMPROBACIÓN:

```
--- MENÚ ---
1. Crear producto y añadirlo a la lista
2. Guardar el inventario
3. Salir
Opción: 1
Nombre: SARDINAS
Precio: 20
Cantidad: 4
Producto añadido
--- MENÚ ---
1. Crear producto y añadirlo a la lista
2. Guardar el inventario
3. Salir
Opción: 1
Nombre: PAN DE PIPAS
Precio: 2
Cantidad: 50
Producto añadido
--- MENÚ ---
1. Crear producto y añadirlo a la lista
2. Guardar el inventario
3. Salir
Opción: 2
--- MENÚ ---
1. Crear producto y añadirlo a la lista
2. Guardar el inventario
3. Salir
Opción: 3
PS C:\Users\CampusFP\Desktop\GUARDAR COSAS PYTHON> |
```

