



INFORMÁTICA Y
TELECOMUNICACIONES

Fundamentos de Programación FPY

Archivos en Python

3.3.1: Contenidos

01

Introducción a los
archivos

02

Abrir y cerrar
archivos

03

Operación básica

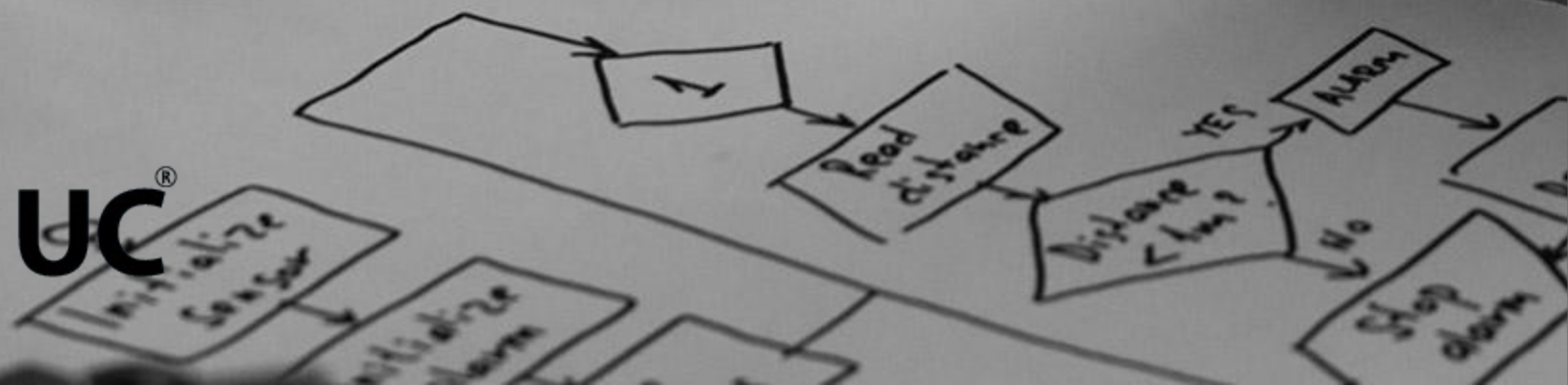
04

Reflexiones



01

Introducción a los archivos



Archivos en Python



En Python, un lenguaje versátil y ampliamente utilizado, trabajar con archivos es esencial en diversas situaciones.

Situaciones comunes que requieren la manipulación de archivos incluyen el almacenamiento de configuraciones, la lectura de datos de entrada, la escritura de registros de salida y la gestión de información estructurada en formatos como TXT, CSV o JSON

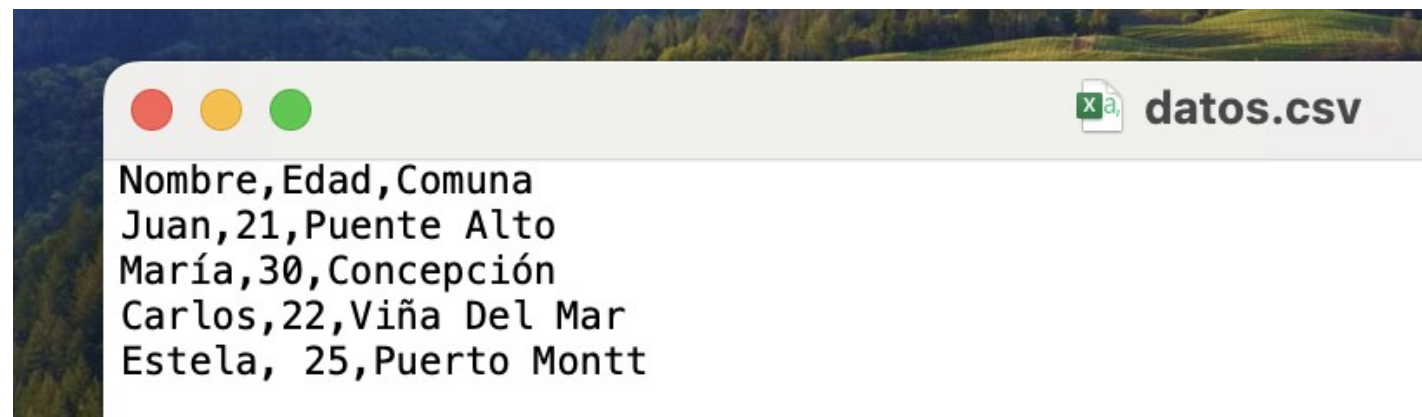
Archivos en Python



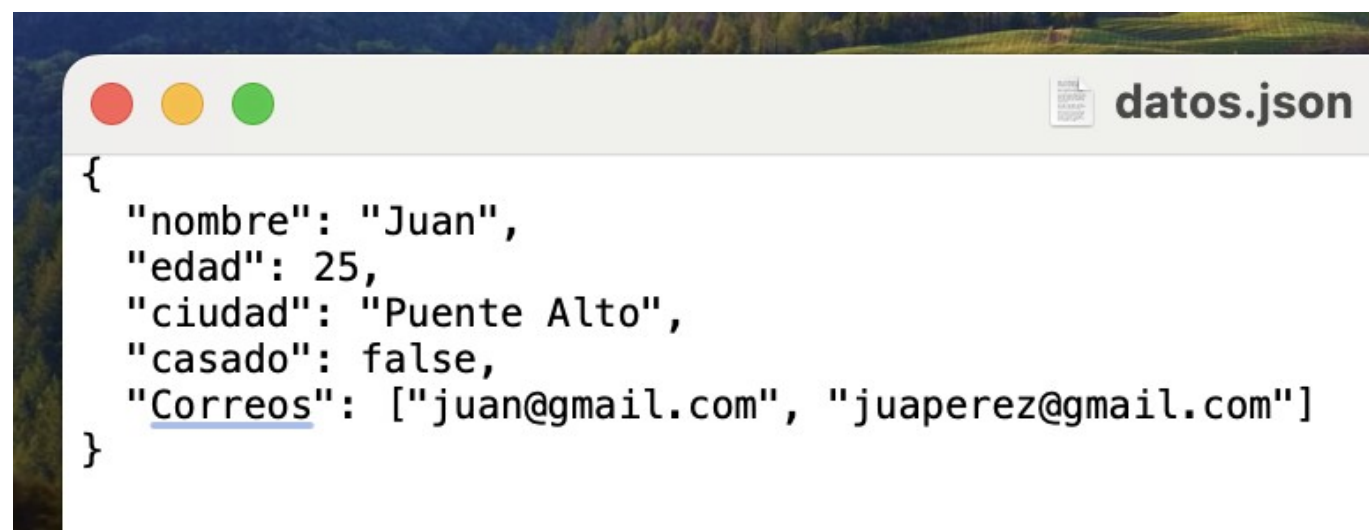
En el ámbito de la programación con Python, el uso de archivos en formatos como CSV, JSON y TXT desempeña un papel crucial. Al persistir datos en archivos, se logra preservar la información entre ejecuciones del programa, facilitando la interoperabilidad con otros sistemas y permitiendo a los usuarios ajustar configuraciones sin modificar el código fuente. Además, estos formatos ofrecen una estructura **legible y versátil para intercambiar datos**, compartir información entre aplicaciones y respaldar la información de manera eficiente. Este enfoque se traduce en **flexibilidad**, mantenimiento sencillo y un manejo eficaz de grandes conjuntos de datos.

Archivos en Python

En informática, es muy común utilizar archivos de tipo TXT, CSV o JSON, explicaremos que significan estos tipos de archivos:



```
Nombre,Edad,Comuna
Juan,21,Puente Alto
María,30,Concepción
Carlos,22,Viña Del Mar
Estela, 25,Puerto Montt
```



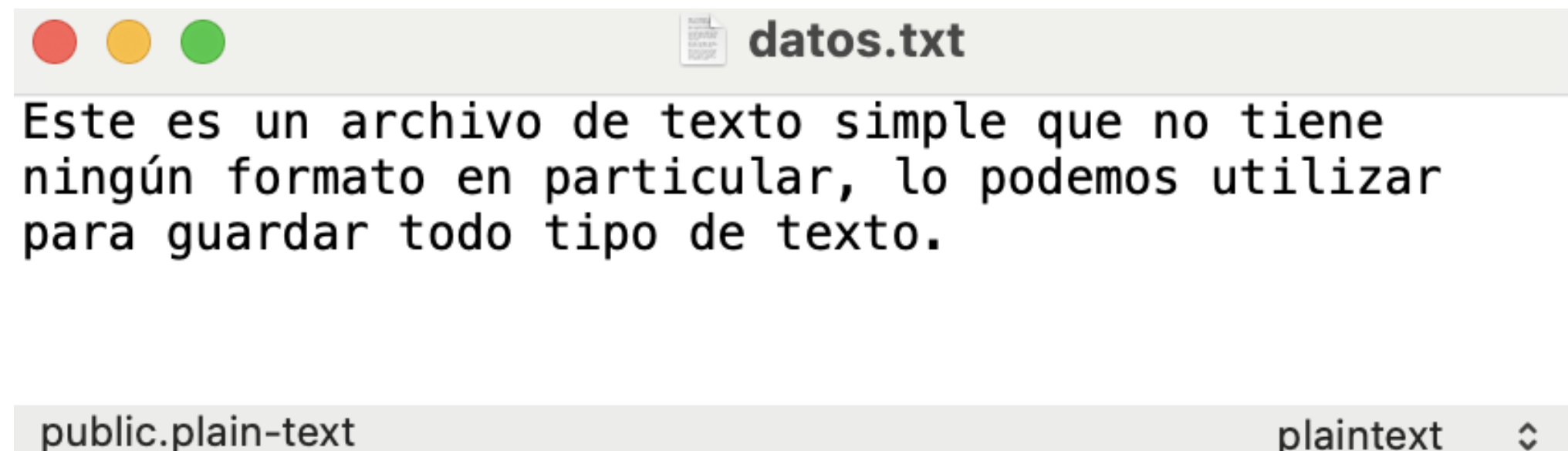
```
{
  "nombre": "Juan",
  "edad": 25,
  "ciudad": "Puente Alto",
  "casado": false,
  "Correos": ["juan@gmail.com", "juaperez@gmail.com"]
}
```

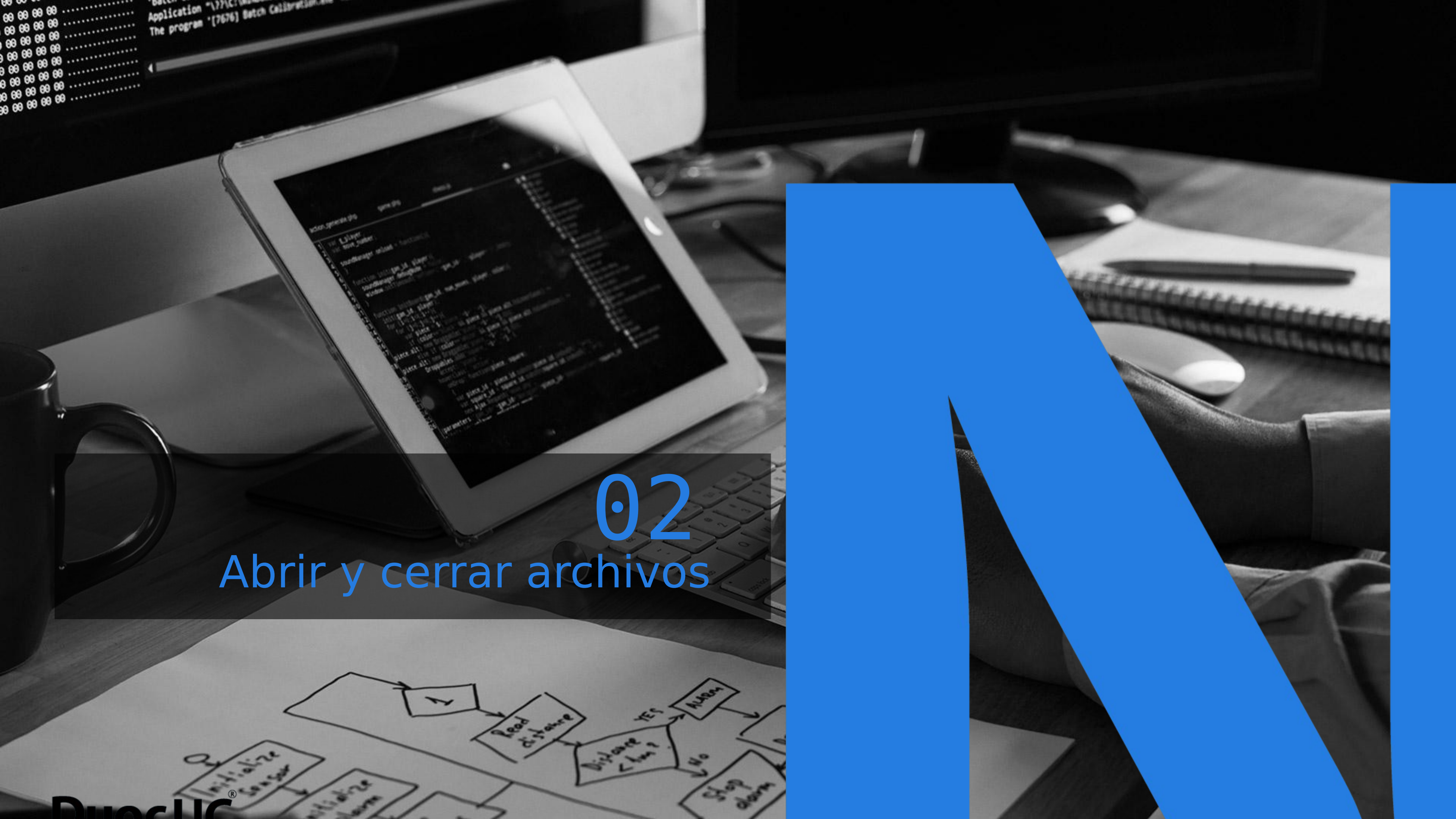
CSV: Un archivo CSV (Comma-Separated Values) es un formato de archivo que se utiliza para almacenar datos tabulares (datos en forma de tabla) en un formato de texto plano. En un archivo CSV, cada línea del archivo representa una fila de datos, y las columnas están separadas por un carácter delimitador, comúnmente una coma (,).

JSON: (JavaScript Object Notation) es un formato de intercambio de datos ligero y fácil de leer, que se utiliza para la transmisión de datos entre un servidor y una aplicación web, así como para el almacenamiento y el intercambio de datos.

Archivos en Python

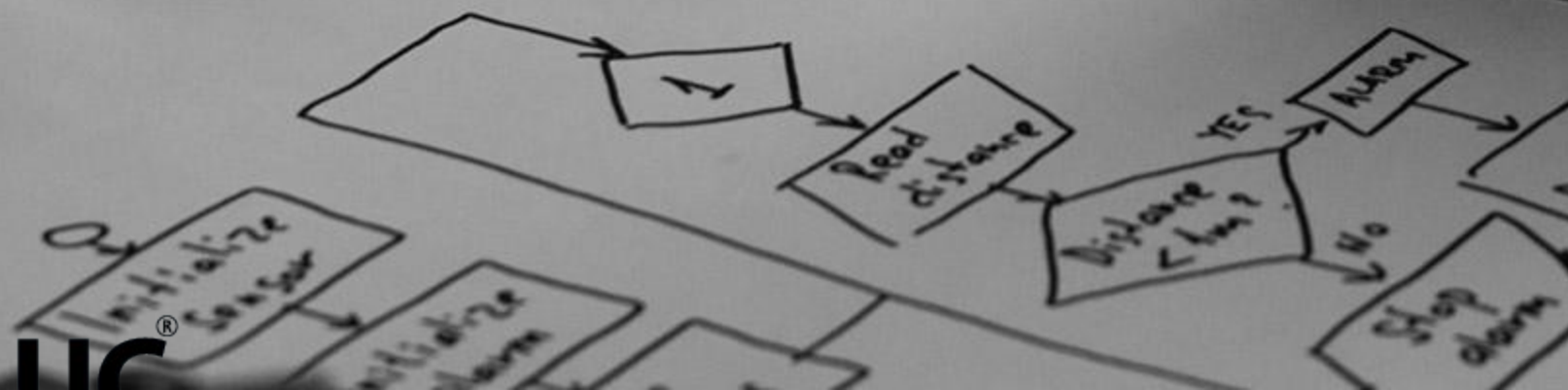
TXT: Tipo de archivo muy utilizado para guardar todo tipo de textos. TXT proviene de la palabra Text.





02

Abrir y cerrar archivos



Crear archivos TXT

```
#crear archivo, w es permiso de escritura
#Las comillas al inicio y 3 al final del texto representan un
texto
#con saltos de línea
datos = """Este es un archivo de texto simple que no tiene
ningún formato en particular, lo podemos utilizar
para guardar todo tipo de texto.
"""

with open('archivo.txt', 'w') as archivo:
    archivo.write(datos)
```

Abrir archivos TXT



Opción 1

Permisos: 'r' (lectura), 'w' (escritura), 'r+' (lectura/escritura)

```
archivo = open('datos.txt', 'r')
    contenido = archivo.read()
print(contenido)
archivo.close()
```

Opción 2

Usando el contexto 'with', el archivo se cierra automáticamente al salir del bloque 'with'

```
with open('datos.txt', 'r') as archivo:
    contenido = archivo.read()
print(contenido)
```

The screenshot shows a code editor with two tabs: 'archivotxt.py' and 'datos.txt'. The 'datos.txt' tab is active and contains the following text:

```
1 Este es un archivo de texto simple que no tiene
2 ningún formato en particular, lo podemos utilizar
3 para guardar todo tipo de texto.
```

Escribir y Leer archivos CSV

```
import csv
# permiso w es escritura
with open('nuevo_archivo.csv', 'w', newline='') as
archivo_csv:
    escritor_csv = csv.writer(archivo_csv)
    # Escribir una fila en el archivo CSV
    escritor_csv.writerow(['Nombre', 'Edad', 'Comuna'])
    # Escribir múltiples filas en el archivo CSV
    escritor_csv.writerows([
        ['Esteban', 25, 'Santiago'],
        ['María', 30, 'Valparaíso'],
        ['Carlos', 22, 'Osorno'],
        ['Sigrid', 25, 'Santiago'],
        ['Daniela', 30, 'La Cisterna'],
        ['Aylen', 22, 'La florida']
    ])
```



Se crea un archivo con extensión CSV

```
archivotxt.py • archivocsv.py • nuevo_archivo.csv ×
nuevo_archivo.csv
1 Nombre,Edad,Comuna
2 Esteban,25,Santiago
3 María,30,Valparaíso
4 Carlos,22,Osorno
5 Sigrid,25,Santiago
6 Daniela,30,La Cisterna
7 Aylen,22,La florida
```

```
# Leer Archivos CSV
import csv
# Sintaxis: open('nombre_del_archivo.csv', 'modo',
newline='')
# Modo común: 'r' (lectura)
with open('nuevo_archivo.csv', 'r', newline='') as
archivo_csv:
    lector_csv = csv.reader(archivo_csv)
    for fila in lector_csv:
        print(fila)
```

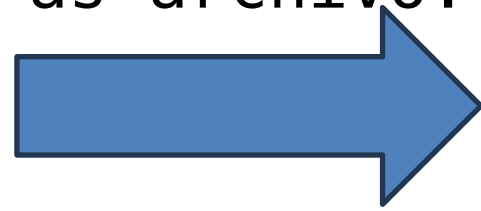
```
['Nombre', 'Edad', 'Comuna']
['Esteban', '25', 'Santiago']
['María', '30', 'Valparaíso']
['Carlos', '22', 'Osorno']
['Sigrid', '25', 'Santiago']
['Daniela', '30', 'La Cisterna']
['Aylen', '22', 'La florida']
```


Escribir y Leer archivos JSON

```
import json
# Datos JSON
datos = {
    "nombre": "Esteban",
    "edad": 25,
    "comuna": "Santiago",
    "estudios": ["colegio Arturo Prat", "liceo el bosque",
    "Duoc UC", "Diplomado Duoc UC"]
}
```

Se crea un archivo con extensión JSON

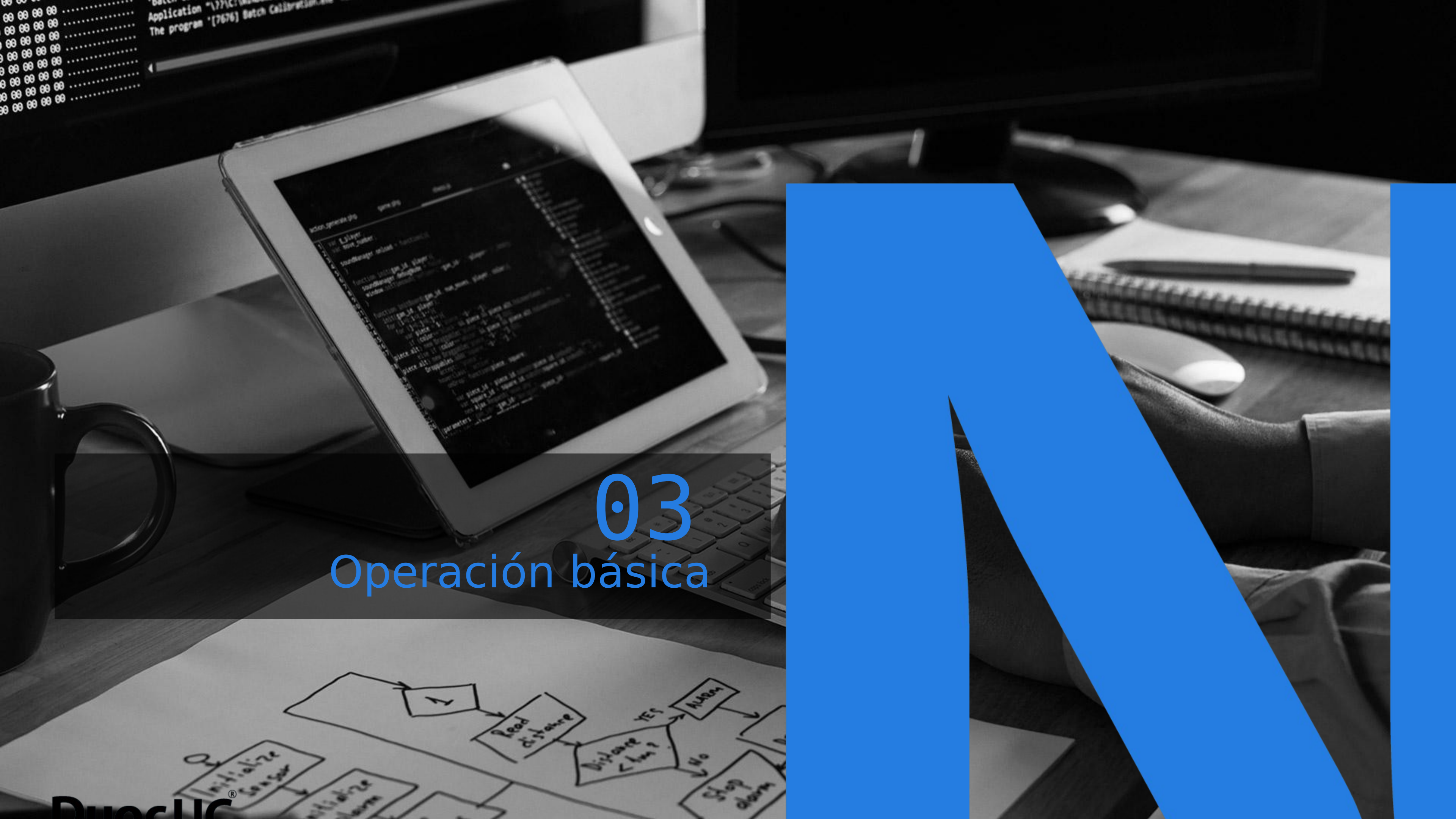
```
# Abre el archivo, w es escritura
with open('archivo.json', 'w') as archivo:
    json.dump(datos, archivo)
```



```
archivojson.py • {} archivo.json ×
{} archivo.json > ...
1 {"nombre": "Esteban", "edad": 25,
2  "comuna": "Santiago",
3  "estudios": ["colegio Arturo Prat", "liceo el bosque",
4  "Duoc UC", "Diplomado Duoc UC"]}
```

```
import json
```

```
# Abrir archivo, r es permiso de lectura
with open('archivo.json', 'r') as archivo:
    datos_leidos = json.load(archivo)
    print(datos_leidos)
```

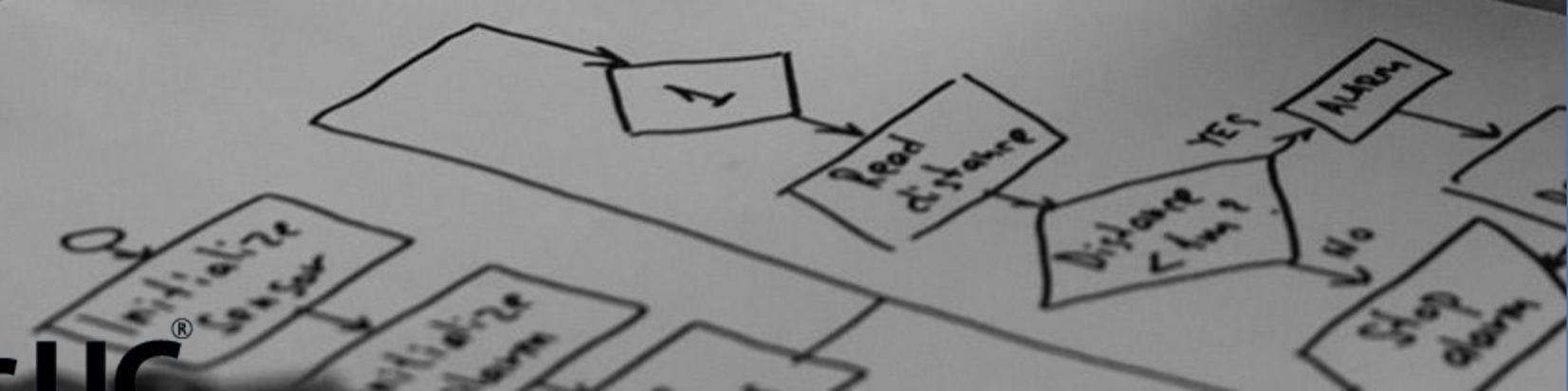


00 00 00 00
00 00 00 00
00 00 00 00
00 00 00 00
00 00 00 00
00 00 00 00
00 00 00 00
00 00 00 00
00 00 00 00
00 00 00 00

batch: ...
Application "C:\Program Files\..."
The program '[7676] Batch Calibration...'

03

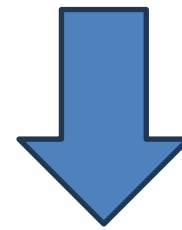
Operación básica



¿Cómo leer el contenido y realizar acciones?

Realicemos este ejercicio. ¿Qué ocurre si queremos leer y realizar alguna acción con nuestros datos, por ejemplo, determinar si los usuarios son mayores o menores de edad?

```
import csv
with open('datos.csv', 'r') as archivo_csv:
    lector_csv = csv.DictReader(archivo_csv)
    # recorremos cada fila con un For
    for fila in lector_csv:
        nombre = fila['Nombre']
        edad = int(fila['Edad'])
        comuna = fila['Comuna']
        estado_edad = "Mayor de Edad" if edad >= 18 else "Menor de Edad"
        print(f"{nombre} tiene {edad} años, es {estado_edad} y vive en {comuna}")
```



PROBLEMAS	SALIDA	CONSOLA DE DEPURACIÓN	TERMINAL	PUERTO
Juan tiene 21 años, es Mayor de Edad y vive en Puente Alto				
María tiene 30 años, es Mayor de Edad y vive en Concepción				
Carlos tiene 22 años, es Mayor de Edad y vive en Viña Del Mar				
Estela tiene 25 años, es Mayor de Edad y vive en Puerto Montt				

Reflexionemos



- Debate con tu docente: ¿Por qué es recomendable utilizar archivos de tipo TXT, CSV o JSON para la transferencia o utilización de datos, y no un Excel o correo electrónico?
- ¿En tu dispositivo celular guardas notas? ¿crees que empresas externas tienen acceso a esta información que guardas? ¿Cómo se obtiene?
- ¿Cómo funciona una transferencia de dinero?, ¿será una instrucción enviada por medio de un archivo de texto, JSON o similar por donde viaja tu dinero?
- Cuando un comercio genera una boleta o factura electrónica, y debe informar al Servicio de Impuestos Internos, ¿los sistemas envían un archivo de texto, JSON o similar?