Trabajar con archivos y CSV en Python

1. Reading files

Este script lee el contenido de un archivo de texto y lo imprime en la consola.

```
# Leer un archivo de texto
with open('example.txt', 'r') as file:
    content = file.read()
    print(content)
```

2. Iterating through files

Este script itera a través de cada línea de un archivo de texto e imprime cada línea.

```
# Iterar a través de las líneas de un archivo de texto
with open('example.txt', 'r') as file:
    for line in file:
        print(line.strip())
```

3. Writing files

Este script escribe texto en un archivo, creando el archivo si no existe.

```
# Escribir en un archivo de texto
with open('output.txt', 'w') as file:
    file.write('Hello, World!\n')
    file.write('This is a new line.')
```

4. File paths

Este script muestra cómo manejar rutas de archivos utilizando el módulo os.

```
import os

# Definir y mostrar la ruta del archivo
file_path = os.path.join('folder', 'example.txt')
print(f'The file path is: {file_path}')
```

5. How to write file paths in code

Este script muestra cómo escribir rutas de archivos en el código de manera segura.

```
import os

# Definir una ruta de archivo utilizando rutas absolutas y relativas
absolute_path = os.path.abspath('example.txt')
relative_path = 'folder/example.txt'

print(f'Absolute path: {absolute_path}')
print(f'Relative path: {relative_path}')
```

6. Working with Files

Este script demuestra cómo realizar operaciones básicas con archivos, como verificar si el archivo existe.

```
import os

# Verificar si el archivo existe
file_path = 'example.txt'
if os.path.exists(file_path):
    print(f'The file {file_path} exists.')
else:
```

```
    print(f'The file {file_path} does not exist.')
```

7. More file Iinformation

Este script muestra cómo obtener información adicional sobre un archivo, como su tamaño y fecha de modificación.

```
import os
import time

# Obtener información del archivo
file_path = 'example.txt'
file_info = os.stat(file_path)

print(f'Size: {file_info.st_size} bytes')
print(f'Last modified: {time.ctime(file_info.st_mtime)}')
```

8. Directories

Este script muestra cómo crear y listar directorios.

```
import os

# Crear un nuevo directorio
os.makedirs('new_folder', exist_ok=True)

# Listar directorios en el directorio actual
for item in os.listdir('.'):
    if os.path.isdir(item):
        print(f'Directory: {item}')
```

9. What is a CSV file?

Un archivo CSV (Comma-Separated Values) es un formato de archivo que utiliza comas para separar valores.

```
# Ejemplo de datos en formato CSV
csv_data = "name,age,city\nJohn,30,New York\nJane,25,Los Angeles"
```

```
print(csv_data)
```

10. Reading CSV files

Este script lee un archivo CSV y muestra su contenido.

```
import csv

# Leer un archivo CSV
with open('example.csv', 'r') as file:
    reader = csv.reader(file)
    for row in reader:
        print(row)
```

11. Generating CSV

Este script genera un archivo CSV a partir de una lista de datos.

```
import csv

# Datos para el archivo CSV
data = [
    ["name", "age", "city"],
    ["John", 30, "New York"],
    ["Jane", 25, "Los Angeles"]
]

# Escribir en un archivo CSV
with open('output.csv', 'w', newline='') as file:
    writer = csv.writer(file)
    writer.writerows(data)
```

12. Reading and writing CSV files with dictionaries

Este script lee y escribe archivos CSV utilizando diccionarios para mayor claridad.

```
import csv

# Leer un archivo CSV usando diccionarios
with open('example.csv', 'r') as file:
    reader = csv.DictReader(file)
    for row in reader:
        print(row)

# Escribir en un archivo CSV usando diccionarios
fieldnames = ['name', 'age', 'city']
data = [
    {'name': 'John', 'age': 30, 'city': 'New York'},
    {'name': 'Jane', 'age': 25, 'city': 'Los Angeles'}
]

with open('output_dict.csv', 'w', newline='') as file:
    writer = csv.DictWriter(file, fieldnames=fieldnames)
    writer.writeheader()
    writer.writerows(data)
```