

```
with open(archivo, ' ', encoding='utf8') as archivo:
w = write      r = read  a = append

lista_str = ['Hola, como estas?\n', 'Me llamo Roberto, y vos?\n']
with open( 'ejemploArchivoOut.txt', 'w' ) as archivo :
    archivo.writelines(lista_str)
```

```
import matplotlib.pyplot as plt
plt.plot(x, y, linestyle , color = "green", linewidth = 3) →
DATOS Y PERSONALIZACIÓN

plot = ;
plt.bar →BARRAS
plt.hist → HISTOGRAMA
plt.scatter → SCATTERPLOTS

linestyle =;
"--" → Punteados
"_^" → Con triángulos

plt.figure()
plt.plot(claves1, valores1, label="nombre1") → DATOS1
plt.plot(claves2, valores2, label="nombre2") → DATOS2
plt.legend(loc='upper left')
plt.xlabel('Turnos')
plt.ylabel('Puntos')
plt.title('Gráfico del juego 10.000!')
plt.show()
```

si quiero copiar todo un archivo igual a otro:

```
def copiar(origen, destino):
with open(origen, "r", encoding='utf8') as archivo:
    contenido = archivo.read()
with open(destino , "w", encoding='utf8') as archivo:
    archivo.write(contenido)
```

mismo archivo y reemplazar palabras

```
def sed(nombre, original , nuevo):
with open(nombre, "r", encoding="utf-8") as archivo:
    contenido = archivo.read()
    contenido = contenido.replace(original , nuevo)
with open(nombre, "w", encoding="utf-8") as archivo:
    archivo.write(contenido)
```

ordenar un archivo csv según categoría

recibe lista de datos y retorna cvs

```
def guardar_csv(nombre, tabla):
with open(nombre, "w") as archivo:
    for fila in tabla:
        linea = ",".join(fila) + "\n"
        archivo.write(linea)
```

```
def ordenar(nombre, categoria):
    with open(nombre, 'r', encoding='utf8') as archivo:
        lineas = archivo.readlines()

    titulos = lineas[0].strip().split(',')
    dic = {}
    for titulo in titulos:
        dic[titulo] = []

    for linea in lineas[1:]:
        valores = linea.strip().split(',')
        for clave, valor in zip(titulos, valores):
            dic[clave].append(valor)

    sorteador = dic[categoria]
    ordenado = sorted(zip(sorteador, range(0, len(sorteador))))

    nombre_nuevo = nombre[:-4] + '_ordenado_por_' + categoria + '.csv'
    with open(nombre_nuevo, 'w', encoding='utf8') as archivo:
        archivo.write(','.join(titulos) + '\n')
        for _, data in ordenado:
            texto = ''
            for titulo in titulos:
                texto += dic[titulo][data] + ','
            texto = texto[:-1] + '\n'
            archivo.write(texto)

ordenar('gente.csv', 'PAIS')
```

cifrar un archivo normal

```
# Genero dos diccionarios: uno para convertir de letra a numero y otro inverso
# para convertir de número a letra.
abecedario = "abcdefghijklmnopqrstuvwxyz" # secuencia de letras (minúsculas)
numeros = range(len(abecedario))         # creo lista de números 0 al 25
letra_num = dict(zip(abecedario, numeros)) # letra_num["a"] -> 0
num_letra = dict(zip(numeros, abecedario)) # num_letra[2] -> "c"

def cifrar(origen, destino):
    """
    Cifra el contenido de un archivo en otro utilizando un código secreto.

    Entradas:
    - origen (str): Nombre del archivo cuyo contenido se quiere cifrar.
    - destino (str): Nombre del archivo donde se guardará el texto cifrado.
      Si el archivo no existe, lo crea, y si existe, lo sobrescribe.
    """
    with open(origen, "r", encoding="utf-8") as archivo:
        contenido = archivo.read() # leo el contenido del archivo de origen
        contenido = list(contenido) # convierto a lista para hacerlo mutable
        for i in range(len(contenido)): # recorro el contenido del archivo
            char = contenido[i].lower() # leo el próx. caracter (en minúscula)
            if char in abecedario: # si es una letra...
                num = (letra_num[char] + 13) % 26 # obtengo el número codificado
                contenido[i] = num_letra[num] # reemplazo el valor
        contenido = "".join(contenido) # vuelvo a convertir a str
        with open(destino, "w", encoding="utf-8") as archivo:
            archivo.write(contenido) # escribo en el archivo de destino

# Ejecuto la función
cifrar("ejemplo.txt", "ejemplo_cifrado.txt")
```

<p>$f(x) = x^3$</p> <p>Para x de -100 a 100 con 1001 valores equiespaciados</p> <p>Línea verde</p> <p>Nombres de ejes X e Y respectivamente</p>	<p>$f(x) = x^2$</p> <p>Para x de -5 a 5 con 101 valores equiespaciados</p> <p>Línea punteada de color rojo</p> <p>Nombres de ejes X e Y respectivamente</p>
--	--

<pre> import matplotlib.pyplot as plt vuelta = (100 - -100)/(1001 - 1) X = [-100] Y = [X[-1]**3] for _ in range(11 - 1): X.append(X[-1] + vuelta) Y.append(X[-1]**3) plt.figure() plt.plot(X,Y, color='green') plt.ylabel("Y") plt.xlabel("X") plt.show()</pre>	<pre> import matplotlib.pyplot as plt vuelta2 = (5 + 5)/(101 - 1) X = [-5] Y = [X[-1]**2] for _ in range(101 - 1): X.append(X[-1] + vuelta2) Y.append(X[-1]**2) plt.figure() plt.plot(X,Y,"--", color='red') plt.ylabel("Y") plt.xlabel("X") plt.show()</pre>
--	--

Grafique la lista = [6, 2, 5, 6, 8, 1, 3, 6, 7, 3] como un gráfico de barras donde los valores de la lista son las alturas de cada barra

```

import matplotlib.pyplot as plt

# Defino la lista de alturas
lista = [6, 2, 5, 6, 8, 1, 3, 6, 7, 3]

# Genero el gráfico de barras con plt.bar(). Para poder graficar la lista
# requiero tener los valores de x de cada barra. Para esto utilizo range.
# El primer valor son los valores en x y los segundos los valores en y o las
# alturas
plt.bar(range(len(lista)), lista)
plt.show()
```

El archivo google.csv contiene información sobre la acción de Google. Las columnas del archivo contienen los siguientes datos:

Date: la fecha de la observación

Open: el precio al que abrió esa mañana

High: el precio más alto al que llegó ese día.

Low: el precio más bajo al que llegó ese día.

Close: el precio al que cerró ese día.

En una misma figura, haga un gráfico en donde se muestre la evolución día a día del precio de apertura, cierre, máximo y mínimo de la acción. Agregue una leyenda para poder identificar cual es cada línea.