```
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 (minusculas)
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")
```

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

```
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()
```

```
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()
```

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

```python
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.

```
import matplotlib.pyplot as plt

# Creo el diccionario que va a guardar las listas de cada columna
datos = {}

# Abro el archivo
with open('google.csv', 'r') as file:
    # Como es un csv, la primer línea contiene el nombre de cada columna. Voy
    # a leerla, limpiarla y separarla con las comas
    linea0 = file.readline().strip().split(',')

    # Ahora que tengo los nombres puedo llenar el diccionario con listas vacias
    # para cada columna
    for valor in linea0:
        datos[valor] = []

    # Ahora recorro el resto de las líneas
    for linea in file:
        # Separo los valores
```

```
        separado = linea.strip().split(',')
        # Ahora recorro tanto los nombres de las columnas como los valores de
        # la línea que acabo de separar, y voy llenando el diccionario con valores
        for llave, valor in zip(datos.keys(), separado):

            # Si el dato no es una fecha, entonces lo convierto a float
            if llave != 'Date':
                valor = float(valor)
            datos[llave].append(valor)


# Lista con los nombres de las columnas a graficar
columns = ['Open','High','Low','Close']

# Grafico con un plt.plot(). Los valores en x son la columna de fechas, los
# valores en y son las columnas. Con label defino los nombres de cada línea
# para que luego con el plt.legend() aparezcan
for tipo in columns:
    plt.plot(datos['Date'],datos[tipo], label = tipo)
plt.legend()

plt.show()
```

El archivo tournaments_2000-2009.csv contiene información sobre los torneos de tenis a nivel ATP jugados en la década del 2000. Grafique en distintas figuras las siguientes consultas:

- **Gráfico de torta: distribución de superficies por torneos jugados (Hard, Clay, Grass, Carpet).**
- **Gráfico de barras: top 10 jugadores con más torneos ganados.**
- **Gráfico scatter: top 10 jugadores con torneos ganados.**
- **Gráfico de barras: top 10 jugadores con mas dinero ganado en total Se espera que para cada gráfico utilice distintos colores, muestre leyendas y título de lo que se está mostrando.**

```python
import matplotlib.pyplot as plt

datos = {}

# Abro el archivo
with open('tournaments_2000-2009.csv', 'r') as file:
    # Como es un csv, la primer línea contiene el nombre de cada columna. Voy
    # a leerla, limpiarla y separarla con las comas
    linea0 = file.readline().strip().split(',')

    # Ahora que tengo los nombres puedo llenar el diccionario con listas vacias
    # para cada columna
    for valor in linea0:
        datos[valor] = []

    # Ahora recorro el resto de las líneas
    for linea in file:
        # Separo los valores
        separado = linea.strip().split(',')

        # Si no hay dinero ganado no cuento la linea
        if separado[linea0.index('prize')] == '':
            continue

        # Ahora recorro tanto los nombres de las columnas como los valores de
        # la línea que acabo de separar, y voy llenando el diccionario con valores
        for llave, valor in zip(datos.keys(), separado):
            # Si el dato es el premio lo paso a int
            if llave == 'prize':
                valor = int(valor)
            datos[llave].append(valor)

# Cuento cuantas veces aparece cada tipo de cancha
courts = ('Hard', 'Clay', 'Grass', 'Carpet')
cantidad = []
for court in courts:
    cantidad.append(datos['court'].count(court))

# Creo una figura y grafico con .pie los valores
plt.figure()
plt.pie(cantidad, labels = courts)
plt.legend()
plt.title('Distribución de canchas')

# Para el segundo gráfico cuento cuantas veces aparece el nombre de cada
# jugador para saber cuantos torneos ganó

winners = {}
for ganador in datos['winner']:
    if ganador not in winners:
        winners[ganador] = 1
    else:
        winners[ganador] += 1

# Separo el top 10
ordenado = dict(sorted(winners.items(), key=lambda item: item[1]))
values = list(ordenado.values())[-10:]
names = list(ordenado.keys())[-10:]
```

```python
# Grafico con barras
plt.figure()
plt.bar(names, values)
plt.xticks(rotation=90)
plt.xlabel('Nombres')
plt.ylabel('Cantidad de torneos ganados')
plt.title('Torneos Ganados')
plt.tight_layout()

# Grafico con scatter
plt.figure()
plt.scatter(names, values)
plt.xticks(rotation=90)
plt.xlabel('Nombres')
plt.ylabel('Cantidad de torneos ganados')
plt.title('Torneos Ganados')
plt.tight_layout()


# De manera similar ahora sumo el dinero ganado de cada uno
total_ganancias = {}
for player, money in zip(datos['winner'], datos['prize']):
    if player not in total_ganancias:
        total_ganancias[player] = money
    else:
        total_ganancias[player] += money


# ordeno y saco el top 10
ordenado = dict(sorted(total_ganancias.items(), key=lambda item: item[1]))
values = list(ordenado.values())[-10:]
names = list(ordenado.keys())[-10:]

# Grafico
plt.figure()
plt.bar(names, values)
plt.title('Dinero Ganado')
plt.xlabel('Nombre')
plt.ylabel('Total ganado')
plt.xticks(rotation=90)
plt.tight_layout()



plt.show()
```