



Archivos

Programación y Laboratorio I



Importancia de los archivos

- Almacenamiento y Persistencia de Datos.
- Administración de los Datos en la Memoria.
- Adquisición de la Información
- Comunicación Dinámica Externa.











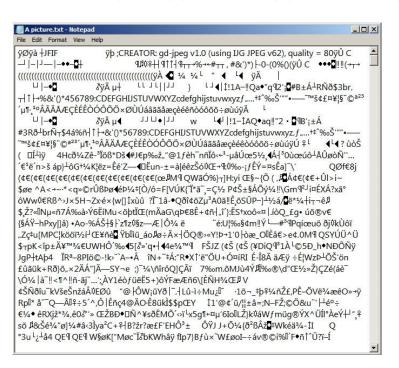




Tipos de Archivos

Archivos Binarios.....

..Archivos de Texto



*CHAMBAAAA.txt: Bloc de notas Archivo Edición Formato Ver Ayuda

Lorem ipsum dolor sit amet consectetur adipiscing elit aliquet dignissim potenti class, eu feugiat dis luctus id viverra consequat ridiculus interdum. Dui sagittis dapibus penatibus mauris porta rhoncus hendrerit conubia, eu in venenatis consequat condimentum volutpat sociis, phasellus vestibulum senectus facilisis dictum vivamus tincidunt. Metus aliquet imperdiet eleifend scelerisque orci neque turpis urna, litora himenaeos nostra hac mi lectus augue, felis vitae dis eu iaculis ad varius.

Dis justo placerat cum dapibus ac vehicula potenti enim sed hendrerit, maecenas felis pharetra pretium bibendum faucibus lectus diam eu sociosqu, habitant purus euismod rhoncus mauris eros et habitasse montes. Pulvinar auctor nullam netus dictum euismod nunc sagittis mauris, sodales fusce vel turpis vestibulum ut bibendum. Lobortis sagittis luctus sem orci cras, aptent auctor etiam duis habitant varius, pellentesque fermentum mi urna.

Morbi mi nec aptent quis est diam conubia nullam fames leo semper bibendum, arcu eu mollis aliquam scelerisque curabitur commodo metus phasellus luctus varius. Congue odio scelerisque cursus porta turpis magna aliquet, pulvinar praesent phasellus diam ultrices proin mattis, fringilla interdum gravida erat suscipit orci. Iaculis ridiculus nibh tellus urna et id etiam magna, ligula pellentesque posuere molestie lacus mi aliquam, commodo volutpat ut netus nam sem inceptos.

Id pharetra augue aenean fames eleifend, praesent phasellus lacinia. Scelerisque at eros nascetur metus torquent curae nisl, habitasse duis feugiat auctor ante volutpat convallis molestie, vivamus sagittis odio ad ullamcorper etiam. Cras placerat bibendum mattis volutpat laoreet ad potenti hac gravida quam justo, penatibus magnis venenatis malesuada ac sed ante quisque sodales molestie, velit montes ornare urna fames phasellus proin tincidunt class iaculis. Mus velit ac pellentesque auctor feugiat luctus sollicitudin a scelerisque, orci ornare mi augue volutpat gravida pulvinar dictumst, potenti sed mattis vulputate quis litora ut justo.

Lacinia metus aptent cum eros dictum hac ad natoque, augue sed nunc litora auctor porta venenatis risus, phasellus mattis libero vulputate elementum dignissim faucibus. Nullam hendrerit dictum cubilia fringilla conque, pulvinar hac pellentesque vestibulum curae mollis, nascetur sociosqu urna posuere. Nibh rutrum interdum cubilia tempor et proin commodo taciti morbi nisl, vehicula suscipit vel integer natoque laoreet tristique eu a, sagittis lobortis tincidunt ante sapien dui consequat ut tempus. Lobortis natoque nibh faucibus id ullamcorper venenatis urna cubilia eget rutrum, himenaeos rhoncus imperdiet phasellus felis potenti nunc fusce facilisis porttitor in, laoreet velit penatibus blandit cras donec fringilla cod mi Línea 9. columna 731 100% Windows (CRLF)



Archivos de texto

Los **archivos de texto** contienen caracteres legibles, es posible no solo leer dicho contenido sino también modificarlo usando un editor de texto.









Archivos binarios

Un **archivo binario** es cualquier archivo que no se pueda interpretar en forma de texto: una imagen, un sonido o incluso un archivo comprimido.









Modos de apertura

MODO	DESCRIPCIÓN
r	Abre un archivo de texto sólo para lectura
rb	Abre un archivo binario sólo para lectura
r+	Abre un archivo de texto para escritura y lectura
W	Abre un archivo de texto sólo para escritura (si existe lo sobreescribe)
wb	Abre un archivo sólo para escritura (si existe lo sobreescribe)
W+	Abre un archivo de texto para escritura y lectura (si existe lo sobreescribe)
а	Abre un archivo para anexar información



Modos de apertura: 'r' 'r+'

r -> abre un archivo sólo para lectura. El puntero al archivo está localizado al comienzo del archivo. Este es el modo predeterminado de la función.

r+ -> abre un archivo para escritura y lectura. El puntero del archivo está localizado al comienzo del archivo.



Modos de apertura: 'w' 'w+'

w -> abre un archivo solo para escritura. Sobreescribe el archivo si este ya existe. Si el archivo no existe lo crea.

w+ -> abre un archivo para escritura y lectura. Sobreescribe el archivo si este ya existe. Si el archivo no existe lo crea.



Modos de apertura: 'a'

a -> abre un archivo para anexo. El puntero del archivo está al final del archivo si este existe. Es decir, el archivo está en modo anexo. Si el archivo no existe, crea un nuevo archivo para escritura.

```
datos: Bloc de notas

Mensaje = "Hola!!"

File = open("datos.txt", "a")

file.write(mensaje)

file.close()

datos: Bloc de notas

Archivo Edición Formato Ver Ayuda

Hola!!
```



Abrir un archivo: open

Para abrir un archivo con Python, podemos usar la función open.

```
archivo = open(nombre_archivo, modo)
```

archivo objeto file, el cual será utilizado para llamar a otros métodos asociados con el archivo.

nombre_archivo string que contiene el nombre del archivo al que queremos acceder.

modo string que contiene el modo de apertura del archivo.



Cerrar un archivo: close

Para cerrar un archivo con Python, podemos usar la función close.

archivo.close()

archivo objeto file que fue obtenido con el método open.



El objeto: file

Una vez que un archivo está abierto y se obtiene el objeto file permite no sólo operar con él sino también obtener mucha información relacionada con ese archivo.



Ejemplo

```
archivo = open(nombre archivo, modo)
```

archivo.closed: retorna **True** si el archivo está cerrado, si no, **False**.

archivo.mode: retorna el modo de acceso con el que el archivo se abrio.

archivo.name: retorna el nombre del archivo.



Leer un archivo: read

El método read permite extraer un string que contenga todos los carácteres del archivo.

```
# Abrimos el archivo en modo lectura y escritura
archivo = open('archivo.txt', 'r+')
texto = archivo.read()
print('El contenido del archivo es: ' + texto)
# Cerramos el archivo
archivo.close()
```



Leer un archivo: read

Es posible limitar al método read a que lea cierta cantidad de bytes read(cantidad)

```
# Abrimos el archivo en modo lectura y escritura
archivo = open('archivo.txt', 'r+')
archivo.seek(100)
texto = archivo.read(10)
print('El contenido del archivo es: ' + texto)
# Cerramos el archivo
archivo.close()
```



Leer un archivo: readlines

Permite obtener una lista con todas las líneas que contiene el archivo.

```
archivo = open('archivo.txt', 'r+')
for linea in archivo.readlines():
    print(linea, end="")
# Cerramos el archivo
archivo.close()
```



Leer un archivo: file

Si solamente requerimos recorrer el archivo línea por línea, el objeto file es iterable.

```
archivo = open('archivo.txt', 'r+')
for linea in archivo:
    print(linea, end="")
# Cerramos el archivo
archivo.close()
```



Leer un archivo: readline

Utilizando la función **readline**() es posible leer desde la posición actual del puntero del archivo hasta el final de la línea y luego posicionar el puntero al comienzo de la siguiente línea.

```
archivo = open('archivo.txt', 'r+')
print(archivo.tell()) #Indica en que byte esta el puntero 0
linea = archivo.readline()
print(linea,end="") # Hola mundo
print(archivo.tell()) #Indica en que byte esta el puntero 11
# Cerramos el archivo
archivo.close()
```



Escribir: write

Escribe una cadena de caracteres dentro del archivo.

```
archivo = open('archivo.txt', 'w')
archivo.write('Primer linea de texto\n')
archivo.write('segunda linea\n')
archivo.write('tercera linea\n')
archivo.close()
```



Escribir: writelines

Escribe una lista de cadena de caracteres dentro del archivo. El parámetro que recibe el método podrá ser cualquier iterable.



Mover puntero: seek

El método seek permite modificar la posición actual del puntero.

```
archivo = open('archivo.txt', 'r+')
archivo.seek(11)
print(archivo.tell()) #Esta en el byte 11
linea = archivo.readline()
print(linea,end="") # Hola mundo
archivo.close()
```



Administrador de contexto: with

Podemos usar la sentencia **with** para abrir archivos en Python y dejar que el intérprete se encargue de cerrar el mismo.

```
with open('archivo.txt', 'r+') as archivo:
   for linea in archivo:
    print(linea, end="")
```



CSV

El CSV es un tipo de documento que representa los datos de forma parecida a una tabla, es decir, organizando la información en filas y columnas.

Las columnas son separadas, por un signo de puntuación (, ; .) u otro carácter y las diferentes filas suelen separarse por un salto de línea.



JSON

El nombre es un acrónimo de las siglas en inglés de JavaScript Object Notation.

JSON es un formato para el intercambio de datos basado en texto. Por lo que es fácil de leer tanto para una persona como para una máquina.



JSON (Escritura)

El paquete **json** permite traducir un diccionario a formato JSON utilizando el método **dump**.

```
import json
data = {}
data['clientes'] = []
data['clientes'].append({ 'nombre': 'Juan', 'edad': 27})
data['clientes'].append({ 'nombre': 'Ana', 'edad': 26})
with open('data.json', 'w') as file:
   json.dump(data, file, indent=4, ensure ascii=False)
```



JSON (Lectura)

La lectura es similar al proceso de escritura, se debe abrir un archivo y procesar esté utilizando el método **load**.

```
import json
with open('data.json') as file:
    data = json.load(file)
```