[io](https://docs.python.org/3/library/io.html?highlight=io" \l "module-io" \o "io: Herramientas básicas para trabajar con streams.)- Herramientas básicas para trabajar con flujos

**Código fuente:**[Lib / io .py](https://github.com/python/cpython/tree/3.7/Lib/io.py)

Visión general

El módulo proporciona las principales funciones de Python para tratar con varios tipos de E / S. Hay tres tipos principales de E / S: *texto de E / S* , *I binaria de E / S* y la *prima de E / S* . Estas son categorías genéricas, y las distintas tiendas de respaldo se pueden usar para cada una de ellas. Un objeto concreto que pertenece a cualquiera de estas categorías se llama un [objeto de archivo](https://docs.python.org/3/glossary.html#term-file-object) . Otros términos comunes son *flujo* y *objeto similar a un archivo* .[io](https://docs.python.org/3/library/io.html?highlight=io" \l "module-io" \o "io: Herramientas básicas para trabajar con streams.)

Independientemente de su categoría, cada objeto de flujo concreto también tendrá varias capacidades: puede ser de solo lectura, de solo escritura o de lectura y escritura. También puede permitir el acceso aleatorio arbitrario (buscar hacia delante o hacia atrás a cualquier ubicación), o solo acceso secuencial (por ejemplo, en el caso de un zócalo o tubería).

Todas las transmisiones son cuidadosas sobre el tipo de datos que les das. Por ejemplo, dar un [str](https://docs.python.org/3/library/stdtypes.html" \l "str" \o "str)objeto al write()método de una secuencia binaria elevará a TypeError. Así será dando un [bytes](https://docs.python.org/3/library/stdtypes.html" \l "bytes" \o "bytes)objeto alwrite()método de un flujo de texto.

*Cambiado en vers io n 3.3:* Parámetros funcionamiento io ns que se utiliza para elevar ahora plantean , ya que ahora es un alias de .[IOError](https://docs.python.org/3/library/exceptions.html" \l "IOError" \o "IOError)[OSError](https://docs.python.org/3/library/exceptions.html#OSError)[IOError](https://docs.python.org/3/library/exceptions.html#IOError)[OSError](https://docs.python.org/3/library/exceptions.html#OSError)

Texto I / O

La E / S de texto espera y produce [str](https://docs.python.org/3/library/stdtypes.html" \l "str" \o "str)objetos. Esto significa que cada vez que el almacén de respaldo se forma nativa hecha de bytes (tal como en el caso de un archivo), codificación y decodificación de los datos se hace transparente, así como opt io traducción nal de caracteres de nueva línea específicas de la plataforma.

La forma más fácil de crear un flujo de texto es con [open()](https://docs.python.org/3/library/functions.html#open), opt io finalmente especificar una codificación:

f = open("myfile.txt", "r", encoding="utf-8")

Los flujos de texto en memoria también están disponibles como objetos:[StringIO](https://docs.python.org/3/library/io.html?highlight=io#io.StringIO)

f = io.StringIO("some initial text data")

La API de secuencia de texto se describe en detalle en el DOCUMENTACIÓN io n de [TextIOBase](https://docs.python.org/3/library/io.html?highlight=io" \l "io.TextIOBase" \o "io.TextIOBase).

I / O binario

Las E / S binarias (también llamadas *E / S en búfer* ) esperan [objetos similares a bytes](https://docs.python.org/3/glossary.html#term-bytes-like-object) y producen [bytes](https://docs.python.org/3/library/stdtypes.html" \l "bytes" \o "bytes)objetos. No codificación, decodificación, o translat nueva línea io n se lleva a cabo. Esta categoría de flujos se puede utilizar para todo tipo de datos que no sean de texto, y también cuando se desea un control manual sobre el manejo de los datos de texto.

La forma más fácil de crear un flujo binario es con [open()](https://docs.python.org/3/library/functions.html#open)la 'b'de la cadena modo:

f = open("myfile.jpg", "rb")

Los flujos binarios en memoria también están disponibles como objetos:[BytesIO](https://docs.python.org/3/library/io.html?highlight=io#io.BytesIO)

f = io.BytesIO(b"some initial binary data: **\x00\x01**")

La API de flujo binario se describe en detalle en los documentos de .[BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BufferedIOBase" \o "io.BufferedIOBase)

Otros módulos de biblioteca pueden proporcionar adic io maneras nales para crear texto o flujos binarios. Ver [socket.socket.makefile()](https://docs.python.org/3/library/socket.html" \l "socket.socket.makefile" \o "socket.socket.makefile)por ejemplo.

Raw I / O

La E / S *sin* procesar (también denominada *E / S sin búfer* ) se utiliza generalmente como un bloque de construcción de bajo nivel para flujos binarios y de texto; rara vez es útil manipular directamente una secuencia en bruto desde el código de usuario. Sin embargo, puede crear una secuencia sin procesar abriendo un archivo en modo binario con el búfer deshabilitado:

f = open("myfile.jpg", "rb", buffering=0)

La API de flujo sin procesar se describe en detalle en los documentos de .[RawIOBase](https://docs.python.org/3/library/io.html?highlight=io" \l "io.RawIOBase" \o "io.RawIOBase)

Interfaz de módulo de alto nivel

io.**DEFAULT\_BUFFER\_SIZE**

Un int que contiene el tamaño de búfer predeterminado utilizado por las clases de E / S almacenadas en búfer del módulo. [open()](https://docs.python.org/3/library/functions.html#open)utiliza el tamaño del archivo (según lo obtenido por [os.stat()](https://docs.python.org/3/library/os.html" \l "os.stat" \o "os.stat)) si es posible.

io.**open**( *archivo* , *modo = 'r'* , *almacenamiento en búfer = -1* , *codificación = Ninguno* , *errores = Ninguno* , *nueva línea = Ninguna* , *closefd = Verdadero* , *abridor = Ninguno*)

Este es un alias para el orden interna [open()](https://docs.python.org/3/library/functions.html#open)func io n.

*excepto io n*io.**BlockingIOError**

Este es un alias de compatibilidad para el builtin excepto io n.[BlockingIOError](https://docs.python.org/3/library/exceptions.html#BlockingIOError)

*excepto io n*io.**UnsupportedOperation**

Una excepto io n heredar [OSError](https://docs.python.org/3/library/exceptions.html" \l "OSError" \o "OSError)y [ValueError](https://docs.python.org/3/library/exceptions.html" \l "ValueError" \o "ValueError)que se plantea cuando una operación no soportado se llama en una corriente.

Transmisiones en memoria

También es posible utilizar una [str](https://docs.python.org/3/library/stdtypes.html#str)o [objeto bytes similar a](https://docs.python.org/3/glossary.html#term-bytes-like-object) como un archivo para lectura y escritura. Para cadenas se pueden utilizar como un archivo abierto en modo texto. Se puede utilizar como un archivo abierto en modo binario. Ambos proporcionan capacidades completas de lectura y escritura con acceso aleatorio.[StringIO](https://docs.python.org/3/library/io.html?highlight=io#io.StringIO)[BytesIO](https://docs.python.org/3/library/io.html?highlight=io#io.BytesIO)

**Ver también**

[sys](https://docs.python.org/3/library/sys.html#module-sys)

contiene el estándar IO corrientes: [sys.stdin](https://docs.python.org/3/library/sys.html" \l "sys.stdin" \o "sys.stdin), [sys.stdout](https://docs.python.org/3/library/sys.html" \l "sys.stdout" \o "sys.stdout), y [sys.stderr](https://docs.python.org/3/library/sys.html" \l "sys.stderr" \o "sys.stderr).

Jerarquía de clases

El implantar mecanismos io n de I / O arroyos está organizado como una jerarquía de clases. Las primeras[clases base abstractas](https://docs.python.org/3/glossary.html#term-abstract-base-class) (ABC), que se utilizan para especificar las diversas categorías de flujos, y luego las clases concretas que proporcionan las implementaciones de flujos estándar.

**Nota**

Las clases base abstractas también proporcionan por defecto implantar mecanismos io ns de algunos métodos con el fin de ayudar a implantar mecanismos io n de clases de flujos de hormigón. Por ejemplo, proporciona implantar mecanismos no optimizado io ns de y.[BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase)readinto()[readline()](https://docs.python.org/3/library/io.html?highlight=io" \l "io.IOBase.readline" \o "io.IOBase.readline)

En la parte superior de la jerarquía de E / S se encuentra la clase base abstracta . Define la interfaz básica a un stream. Tenga en cuenta, sin embargo, que no hay separat io n entre la lectura y la escritura en los arroyos; implantar mecanismos io ns están autorizados a aumentar si no son compatibles con una determinada operación.[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)[UnsupportedOperation](https://docs.python.org/3/library/io.html?highlight=io#io.UnsupportedOperation)

El abc se extiende . Se trata de la lectura y escritura de bytes a un flujo. subclases para proporcionar una interfaz a los archivos en el sistema de archivos de la máquina.[RawIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.RawIOBase)[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)[FileIO](https://docs.python.org/3/library/io.html?highlight=io#io.FileIO)[RawIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.RawIOBase)

El ABC se ocupa del almacenamiento en búfer en un flujo de bytes sin procesar ( ). Sus subclases, , , y corrientes de amortiguación que son leer, escribir, y ambos lectura y escritura. proporciona una interfaz de búfer para secuencias de acceso aleatorio. Otra subclase,, es un flujo de bytes en memoria.[BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase)[RawIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.RawIOBase)[BufferedWriter](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedWriter)[BufferedReader](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedReader)[BufferedRWPair](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedRWPair)[BufferedRandom](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedRandom)[BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase)[BytesIO](https://docs.python.org/3/library/io.html?highlight=io#io.BytesIO)

El ABC, otra subclase de , trata con secuencias cuyos bytes representan texto, y maneja la codificación y decodificación desde y hacia las cadenas. , que lo amplía, es una interfaz de texto almacenado en búfer a una secuencia sin formato ( ) en búfer . Finalmente, es un flujo en memoria para texto.[TextIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.TextIOBase)[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)[TextIOWrapper](https://docs.python.org/3/library/io.html?highlight=io#io.TextIOWrapper)[BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase)[StringIO](https://docs.python.org/3/library/io.html?highlight=io#io.StringIO)

Nombres de los argumentos no son parte de la ESPECIFICA io n, y sólo los argumentos de [open()](https://docs.python.org/3/library/functions.html#open)están destinados a ser utilizados como argumentos de palabra clave.

La siguiente tabla resume los ABC proporcionados por el módulo:[io](https://docs.python.org/3/library/io.html?highlight=io#module-io)

| **A B C** | **Hereda** | **Métodos del talón** | **Métodos de mezcla y propiedades** |
| --- | --- | --- | --- |
| [IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase) |  | fileno,seekytruncate | close, closed, \_\_enter\_\_, \_\_exit\_\_,flush, isatty, \_\_iter\_\_, \_\_next\_\_,readable, readline, readlines, seekable,tell, writable, Ywritelines |
| [RawIOBase](https://docs.python.org/3/library/io.html?highlight=io" \l "io.RawIOBase" \o "io.RawIOBase) | [IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase) | readinto y write | Métodos heredados , y[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)readreadall |
| [BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase) | [IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase) | detach, read,read1, Ywrite | Métodos heredados , y[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)readintoreadinto1 |
| [TextIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.TextIOBase) | [IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase) | detach, read,readline, Y write | Heredados métodos, , , y[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)encodingerrorsnewlines |

Clases base de E / S

*clase*io.**IOBase**

La clase base abstracta para todas las clases de E / S, actuando en flujos de bytes. No hay constructor público.

Esta clase proporciona implantar mecanismos abstracto vacío io ns para muchos métodos que las clases derivadas pueden anular selectivamente; Las implementaciones predeterminadas representan un archivo que no se puede leer, escribir o buscar.

A pesar de que no declare , o porque sus firmas variarán, implantar mecanismos io ns y los clientes deben considerar los métodos de parte de la interfaz. Además, implantar mecanismos io ns podrá plantear una (o ) cuando las operaciones que no son compatibles son llamados.[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)read()readinto()write()[ValueError](https://docs.python.org/3/library/exceptions.html#ValueError)[UnsupportedOperation](https://docs.python.org/3/library/io.html?highlight=io#io.UnsupportedOperation)

El tipo básico utilizado para leer o escribir datos binarios en un archivo es [bytes](https://docs.python.org/3/library/stdtypes.html#bytes). Otros [objetos similares a bytes también](https://docs.python.org/3/glossary.html#term-bytes-like-object) se aceptan como argumentos de método. En algunos casos, como por ejemplo [readinto()](https://docs.python.org/3/library/io.html?highlight=io" \l "io.RawIOBase.readinto" \o "io.RawIOBase.readinto), un objeto grabable tal como [bytearray](https://docs.python.org/3/library/stdtypes.html" \l "bytearray" \o "bytearray) se requiere. Las clases de E / S de texto funcionan con [str](https://docs.python.org/3/library/stdtypes.html" \l "str" \o "str)datos.

Tenga en cuenta que la llamada a cualquier método (incluso consultas) en un flujo cerrado no está definida. Implantar mecanismos io ns pueden plantear [ValueError](https://docs.python.org/3/library/exceptions.html" \l "ValueError" \o "ValueError)en este caso.

[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)(y sus subclases) admite el protocolo iterador, lo que significa que un objeto se puede iterar sobre el rendimiento de las líneas en una secuencia. Las líneas se definen de forma ligeramente diferente según si la secuencia es una secuencia binaria (bytes de rendimiento) o una secuencia de texto (cadenas de caracteres de rendimiento). Ver abajo.[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)[readline()](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase.readline)

[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)También es un administrador de contexto y por lo tanto apoya la [with](https://docs.python.org/3/reference/compound_stmts.html" \l "with)declaración. En este ejemplo, *el archivo* se cierra después de que el [with](https://docs.python.org/3/reference/compound_stmts.html" \l "with)conjunto de la declaración está terminado, incluso si una excepción io n situaciones:

**with** open('spam.txt', 'w') **as** file:

file.write('Spam and eggs!')

[IOBase](https://docs.python.org/3/library/io.html?highlight=io" \l "io.IOBase" \o "io.IOBase) proporciona estos atributos de datos y métodos:

**close**( )

Enjuague y cierre esta corriente. Este método no tiene efecto si el archivo ya está cerrado. Una vez que el archivo se cierra, cualquier Func io n en el archivo (por ejemplo, lectura o escritura) elevará una [ValueError](https://docs.python.org/3/library/exceptions.html" \l "ValueError" \o "ValueError).

Como conveniencia, se permite llamar a este método más de una vez; Sin embargo, solo la primera llamada tendrá efecto.

**closed**

True si la corriente está cerrada

**fileno**( )

Devuelve el descriptor de archivo subyacente (un entero) de la secuencia si existe. Se [OSError](https://docs.python.org/3/library/exceptions.html" \l "OSError" \o "OSError)genera un error si el objeto IO no utiliza un descriptor de archivo.

**flush**( )

Lave los buffers de escritura de la secuencia si es aplicable. Esto no hace nada para las secuencias de solo lectura y no bloqueantes.

**isatty**( )

Regrese Truesi el flujo es interactivo (es decir, conectado a un terminal / dispositivo tty).

**readable**( )

Volver Truesi la secuencia se puede leer desde. Si False, read() subirá [OSError](https://docs.python.org/3/library/exceptions.html" \l "OSError" \o "OSError).

**readline**( *tamaño = -1*)

Lee y regresa una línea de la corriente. Si se especifica el *tamaño* , como máximo se leerán los bytes de *tamaño* .

El terminador de línea es siempre b'\n'para archivos binarios; para los archivos de texto, el argumento de *nueva línea*[open()](https://docs.python.org/3/library/functions.html#open) se puede usar para seleccionar los terminadores de línea reconocidos.

**readlines**( *pista = -1*)

Lea y devuelva una lista de líneas de la secuencia. se puede especificar una *sugerencia* para controlar el número de líneas leídas: no se leerán más líneas si el tamaño total (en bytes / caracteres) de todas las líneas hasta ahora supera la *sugerencia* .

Tenga en cuenta que ya es posible iterar en objetos de archivo sin llamar .for line in file:...file.readlines()

**seek**( *Desplazamiento*[ , *de donde*] )

Cambiar la corriente postulan io n a lo dado byte de *desplazamiento* . *desplazamiento* se interpreta con relación a la posit io n indica por *dónde* . El valor predeterminado para *whence* es SEEK\_SET. Valores por los *cuales* son:

* SEEK\_SETo 0- inicio de la transmisión (el valor predeterminado); *la compensación* debe ser cero o positiva
* SEEK\_CURo 1- posit corriente actual io n; *la compensación* puede ser negativa
* SEEK\_ENDo 2- final de la corriente; *el desplazamiento* suele ser negativo

Retorno de la nueva absoluta posit io n.

*Nuevo en vers io n 3.1:* Las SEEK\_\*constantes.

*Nuevo en vers io n 3.3:* Algunos sistemas operativos podrían apoyar adic io valores nales, comoos.SEEK\_HOLEo os.SEEK\_DATA. Los valores válidos para un archivo podrían depender de que esté abierto en modo de texto o binario.

**seekable**( )

Devuelve Truesi la transmisión soporta acceso aleatorio. Si False, [seek()](https://docs.python.org/3/library/io.html?highlight=io" \l "io.IOBase.seek" \o "io.IOBase.seek), [tell()](https://docs.python.org/3/library/io.html?highlight=io" \l "io.IOBase.tell" \o "io.IOBase.tell)y [truncate()](https://docs.python.org/3/library/io.html?highlight=io" \l "io.IOBase.truncate" \o "io.IOBase.truncate)elevará [OSError](https://docs.python.org/3/library/exceptions.html" \l "OSError" \o "OSError).

**tell**( )

Devolver la corriente de flujo postulan io n.

**truncate**( *tamaño = ninguno*)

Cambiar el tamaño de la corriente a la dada *tamaño* en bytes (o el posit corriente io n si *el tamaño* no se especifica). La posición actual de la secuencia no se cambia. Este cambio de tamaño puede ampliar o reducir el tamaño del archivo actual. En caso de extensión, el contenido del área de archivo nuevo depende de la plataforma (en la mayoría de los sistemas, los bytes adicionales se rellenan con cero). Se devuelve el nuevo tamaño de archivo.

*Cambiado en vers io n 3.5:* Windows ahora un relleno con ceros archivos cuando se extiende.

**writable**( )

Vuelve Truesi el stream admite escritura. Si False, write()y [truncate()](https://docs.python.org/3/library/io.html?highlight=io" \l "io.IOBase.truncate" \o "io.IOBase.truncate)lo subiremos [OSError](https://docs.python.org/3/library/exceptions.html" \l "OSError" \o "OSError).

**writelines**( *líneas*)

Escribe una lista de líneas a la corriente. Los separadores de línea no se agregan, por lo que es habitual que cada una de las líneas provistas tenga un separador de línea al final.

**\_\_del\_\_**( )

Preparar para el objeto destrucción io n. proporciona un defecto implantar mecanismos io n de este método que llama a la instancia de método.[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)[close()](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase.close)

*clase*io.**RawIOBase**

Clase base para E / S binario sin formato. Se hereda . No hay constructor público.[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)

La E / S binaria sin formato generalmente proporciona acceso de bajo nivel a un dispositivo operativo subyacente o API, y no intenta encapsularlo en primitivas de alto nivel (esto se deja a E / S en búfer y E / S en texto, que se describe más adelante en este página).

En Addit io n a los atributos y métodos de , proporciona los siguientes métodos:[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)[RawIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.RawIOBase)

**read**( *tamaño = -1*)

Lea hasta bytes de *tamaño* del objeto y devuélvalos. Como conveniencia, si el *tamaño* no está especificado o -1, se devuelven todos los bytes hasta EOF. De lo contrario, solo se hace una llamada al sistema. Menos del *tamaño de* bytes pueden ser devueltos si la llamada al sistema operativo devuelve menos de *tamaño* bytes.

Si se devuelven 0 bytes y el *tamaño* no era 0, esto indica el final del archivo. Si el objeto está en modo de no bloqueo y no hay bytes disponibles, Nonese devuelve.

La implantar mecanismos predeterminado io n remite a [readall()](https://docs.python.org/3/library/io.html?highlight=io" \l "io.RawIOBase.readall" \o "io.RawIOBase.readall)y [readinto()](https://docs.python.org/3/library/io.html?highlight=io" \l "io.RawIOBase.readinto" \o "io.RawIOBase.readinto).

**readall**( )

Lea y devuelva todos los bytes de la ruta hasta EOF, utilizando varias llamadas a la ruta si es necesario.

**readinto**( *b*)

Leer bytes en un pre-asignado, se puede escribir [bytes como objeto-](https://docs.python.org/3/glossary.html#term-bytes-like-object) *b* , y devolver el número de bytes leídos. Si el objeto está en modo de no bloqueo y no hay bytes disponibles, Nonese devuelve.

**write**( *b*)

Escriba el [objeto de tipo bytes](https://docs.python.org/3/glossary.html#term-bytes-like-object) dado , *b* , en la secuencia sin procesar subyacente y devuelva el número de bytes escritos. Esto puede ser menor que la longitud de *b* en bytes, dependiendo de las características específicas de la secuencia sin procesar subyacente, y especialmente si está en modo no bloqueante. Nonese devuelve si la secuencia sin procesar se configura para que no se bloquee y no se podría escribir ningún byte en él. La persona que llama puede liberar o mutar *b* después devuelve este método, por lo que el implantar mecanismos io n sólo deben acceder *b* durante la llamada al método.

*clase*io.**BufferedIOBase**

Clase base para flujos binarios que admiten algún tipo de almacenamiento en búfer. Se hereda . No hay constructor público.[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)

La principal diferencia con es que los métodos , y tratará (respectivamente) para leer tanto de entrada como solicitado o para consumir toda la salida dada, a expensas de lo que quizá más de una llamada al sistema.[RawIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.RawIOBase)[read()](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase.read)[readinto()](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase.readinto)[write()](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase.write)

En adic io n, esos métodos se plantean si el flujo prima subyacente está en el modo de no bloqueo y no puede tomar o dar suficientes datos; a diferencia de sus contrapartes, nunca volverán .[BlockingIOError](https://docs.python.org/3/library/exceptions.html" \l "BlockingIOError" \o "BlockingIOError)[RawIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.RawIOBase)None

Además, el [read()](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BufferedIOBase.read" \o "io.BufferedIOBase.read)método no tiene un defecto implantar mecanismos io n que se remite a [readinto()](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BufferedIOBase.readinto" \o "io.BufferedIOBase.readinto).

Un típico implantar mecanismos io n no debe heredar de un implantar mecanismos io n, pero envuelva uno, como y hacer.[BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase)[RawIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.RawIOBase)[BufferedWriter](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedWriter)[BufferedReader](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedReader)

[BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase)proporciona o anula estos métodos y atributo en Addit io n a los de [IOBase](https://docs.python.org/3/library/io.html?highlight=io" \l "io.IOBase" \o "io.IOBase):

**raw**

La secuencia en bruto subyacente (una instancia) que trata. Esto no es parte de la API, y no puede existir en algunos implantar mecanismos io ns.[RawIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.RawIOBase)[BufferedIOBaseBufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase)

**detach**( )

Separe la secuencia sin procesar subyacente del búfer y devuélvala.

Después de que la secuencia sin procesar se haya desconectado, el búfer se encuentra en un estado inutilizable.

Algunos búferes, como , no tienen el concepto de un solo flujo sin procesar para regresar de este método. Ellos suben .[BytesIO](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BytesIO" \o "io.BytesIO)[UnsupportedOperation](https://docs.python.org/3/library/io.html?highlight=io#io.UnsupportedOperation)

*Nuevo en vers io n 3.1.*

**read**( *tamaño = -1*)

Leer y devolver hasta bytes de *tamaño* . Si el argumento se omite, Noneo es negativo, los datos se leen y devuelven hasta que se alcanza EOF. Se [bytes](https://docs.python.org/3/library/stdtypes.html" \l "bytes" \o "bytes)devuelve un objeto vacío si la secuencia ya está en EOF.

Si el argumento es positivo y la secuencia sin formato subyacente no es interactiva, se pueden emitir múltiples lecturas sin procesar para satisfacer el conteo de bytes (a menos que se alcance primero el EOF). Pero para las transmisiones sin procesar interactivas, como máximo se emitirá una lectura en bruto, y un resultado breve no implica que EOF sea inminente.

A se eleva si el flujo bruto subyacente está en modo no bloqueante y no tiene datos disponibles en este momento.[BlockingIOError](https://docs.python.org/3/library/exceptions.html#BlockingIOError)

**read1**( [ *tamaño*] )

Lea y devuelva hasta bytes de *tamaño* , con a lo sumo una llamada al método [read()](https://docs.python.org/3/library/io.html?highlight=io" \l "io.RawIOBase.read" \o "io.RawIOBase.read)(o [readinto()](https://docs.python.org/3/library/io.html?highlight=io" \l "io.RawIOBase.readinto" \o "io.RawIOBase.readinto)) de flujo sin procesar subyacente . Esto puede ser útil si está implementando su propio almacenamiento en búfer sobre un objeto.[BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase)

Si el *tamaño* es -1(el valor predeterminado), se devuelve un número arbitrario de bytes (más de cero a menos que se alcance EOF).

**readinto**( *b*)

Leer bytes en un pre-asignado, se puede escribir [bytes como objeto-](https://docs.python.org/3/glossary.html#term-bytes-like-object) *b* y devolver el número de bytes leídos.

Del mismo modo [read()](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BufferedIOBase.read" \o "io.BufferedIOBase.read), se pueden emitir varias lecturas a la secuencia en bruto subyacente, a menos que esta última sea interactiva.

A se eleva si el flujo bruto subyacente está en modo no bloqueante y no tiene datos disponibles en este momento.[BlockingIOError](https://docs.python.org/3/library/exceptions.html#BlockingIOError)

**readinto1**( *b*)

Lea los bytes en un [objeto](https://docs.python.org/3/glossary.html#term-bytes-like-object) *b* , [similar a un objeto de](https://docs.python.org/3/glossary.html#term-bytes-like-object) escritura preasignado , utilizando como máximo una llamada al método [read()](https://docs.python.org/3/library/io.html?highlight=io" \l "io.RawIOBase.read" \o "io.RawIOBase.read)(o [readinto()](https://docs.python.org/3/library/io.html?highlight=io" \l "io.RawIOBase.readinto" \o "io.RawIOBase.readinto)) de la secuencia sin formato subyacente . Devuelve el número de bytes leídos.

A se eleva si el flujo bruto subyacente está en modo no bloqueante y no tiene datos disponibles en este momento.[BlockingIOError](https://docs.python.org/3/library/exceptions.html#BlockingIOError)

*Nuevo en vers io n 3.5.*

**write**( *b*)

Escriba el [objeto de tipo de bytes](https://docs.python.org/3/glossary.html#term-bytes-like-object) dado , *b* , y devuelva el número de bytes escritos (siempre igual a la longitud de *b* en bytes, ya que si la escritura falla [OSError](https://docs.python.org/3/library/exceptions.html" \l "OSError" \o "OSError), se generará). Dependiendo de la implantar mecanismos real io n, estos bytes pueden escribirse fácilmente a la corriente subyacente, o llevan a cabo en un tampón que por razones de rendimiento y la latencia.

Cuando está en modo de no bloqueo, a se levanta si los datos debían escribirse en el flujo sin procesar pero no podía aceptar todos los datos sin bloquear.[BlockingIOError](https://docs.python.org/3/library/exceptions.html#BlockingIOError)

La persona que llama puede liberar o mutar *b* después devuelve este método, por lo que el implantar mecanismos io n sólo deben acceder *b* durante la llamada al método.

Archivo sin procesar I / O

*clase*( *nombre* , *modo = 'r'* , *closefd = Verdadero* , *abridor = Ninguno*) io.**FileIO**

[FileIO](https://docs.python.org/3/library/io.html?highlight=io#io.FileIO)representa un archivo de nivel de SO que contiene datos de bytes. Implementa la interfaz (y por lo tanto la interfaz, también).[RawIOBase](https://docs.python.org/3/library/io.html?highlight=io" \l "io.RawIOBase" \o "io.RawIOBase)[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)

El *nombre* puede ser una de dos cosas:

* una cadena de caracteres u [bytes](https://docs.python.org/3/library/stdtypes.html" \l "bytes" \o "bytes)objeto que representa la ruta al archivo que se abrirá. En este caso, closefd debe ser True(el valor predeterminado), de lo contrario se generará un error.
* un número entero que representa el número de un descriptor de archivo existente a nivel de sistema operativo al que el objeto resultante dará acceso. Cuando el objeto File IO está cerrado, este fd también se cerrará, a menos que *closefd* esté establecido en .[FileIO](https://docs.python.org/3/library/io.html?highlight=io" \l "io.FileIO" \o "io.FileIO)False

El *modo* puede ser 'r', 'w', 'x'o 'a'para la lectura (por defecto), la escritura, exclusiva creat io n o anexar. El archivo se creará si no existe cuando se abre para escribir o anexar; Se truncará cuando se abra para escribir. [FileExistsError](https://docs.python.org/3/library/exceptions.html" \l "FileExistsError" \o "FileExistsError)se levantará si ya existe cuando se abre para crear. Abrir un archivo para crear implica escritura, por lo que este modo se comporta de una manera similar 'w'. Agregue un '+'al modo para permitir la lectura y escritura simultáneas.

El read()(cuando se llama con un argumento positivo) readinto() y los write()métodos de esta clase solo harán una llamada al sistema.

Se puede utilizar un abridor personalizado pasando un llamador como *abridor* . El descriptor de archivo subyacente para el objeto de archivo se obtiene luego llamando al *abridor* con ( *nombre* , *marcas* ). *abridor*debe devolver un descriptor de fichero abierto (pasando [os.open](https://docs.python.org/3/library/os.html" \l "os.open" \o "os.open)como *abridor* resultados en func io nalidad similar a paso None).

El archivo recién creado [no](https://docs.python.org/3/library/os.html#fd-inheritance) es [heredable](https://docs.python.org/3/library/os.html#fd-inheritance) .

Ver el [open()](https://docs.python.org/3/library/functions.html#open)incorporado en func io n de ejemplos sobre el uso del *abridor de* parámetro.

*Cambiado en vers io n 3.3:* El *abridor* se añadió parámetro. El 'x'modo fue agregado.

*Cambiado en vers io n 3.4:* El archivo está ahora no se puede heredar.

En Addit io n a los atributos y métodos de y , proporciona atributos los siguientes datos:[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)[RawIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.RawIOBase)[FileIO](https://docs.python.org/3/library/io.html?highlight=io#io.FileIO)

**mode**

El modo como se da en el constructor.

**name**

El nombre del archivo. Este es el descriptor de archivo del archivo cuando no se da ningún nombre en el constructor.

Arroyos Buffer

Los flujos de E / S en búfer proporcionan una interfaz de nivel superior a un dispositivo de E / S que la E / S sin formato.

*clase*( [ *initial\_bytes*] ) io.**BytesIO**

A implantar mecanismos corriente io n utilizando una memoria en bytes buffer. Se hereda . El búfer se descarta cuando se llama al método.[BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase)[close()](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase.close)

El opt io argumento nales *initial\_bytes* es un [objeto bytes similar](https://docs.python.org/3/glossary.html#term-bytes-like-object) que contiene los datos iniciales.

[BytesIO](https://docs.python.org/3/library/io.html?highlight=io#io.BytesIO)proporciona o anula estos métodos en Addit io n a los de y :[BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BufferedIOBase" \o "io.BufferedIOBase)[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)

**getbuffer**( )

Devuelva una vista legible y escribible sobre los contenidos del búfer sin copiarlos. Además, si se muta la vista, se actualizarán de forma transparente los contenidos del búfer:

>>>

**>>>** b = io.BytesIO(b"abcdef")

**>>>** view = b.getbuffer()

**>>>** view[2:4] = b"56"

**>>>** b.getvalue()

b'ab56ef'

**Nota**

Mientras exista la vista, el objeto no podrá redimensionarse ni cerrarse.[BytesIO](https://docs.python.org/3/library/io.html?highlight=io#io.BytesIO)

*Nuevo en vers io n 3.2.*

**getvalue**( )

Retorno [bytes](https://docs.python.org/3/library/stdtypes.html" \l "bytes" \o "bytes)conteniendo todo el contenido del buffer.

**read1**( [ *tamaño*] )

En , esto es lo mismo que .[BytesIO](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BytesIO" \o "io.BytesIO)[read()](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase.read)

*Cambiado en vers io n 3.7:* El *tamaño del* argumento se opta ahora por io nal.

**readinto1**( *b*)

En , esto es lo mismo que .[BytesIO](https://docs.python.org/3/library/io.html?highlight=io#io.BytesIO)[readinto()](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase.readinto)

*Nuevo en vers io n 3.5.*

*class*( *raw* , *buffer\_size = DEFAULT\_BUFFER\_SIZE*) io.**BufferedReader**

Un búfer que proporciona acceso de nivel superior a un objeto secuencial legible . Se hereda . Al leer datos de este objeto, es posible que se solicite una mayor cantidad de datos de la secuencia sin procesar subyacente y se mantenga en un búfer interno. Los datos almacenados en búfer se pueden devolver directamente en lecturas posteriores.[RawIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.RawIOBase)[BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase)

El constructor crea una [BufferedReader](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BufferedReader" \o "io.BufferedReader)para la lectura dada *prima* arroyo y *buffer\_size* . Si *buffer\_size* se omite, [DEFAULT\_BUFFER\_SIZE](https://docs.python.org/3/library/io.html?highlight=io" \l "io.DEFAULT_BUFFER_SIZE" \o "io.DEFAULT_BUFFER_SIZE)se utiliza.

[BufferedReader](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedReader)proporciona o anula estos métodos en Addit io n a los de y :[BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BufferedIOBase" \o "io.BufferedIOBase)[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)

**peek**( [ *tamaño*] )

Volver bytes de la transmisión sin avanzar el posit io n. A lo sumo, se realiza una sola lectura en el flujo sin procesar para satisfacer la llamada. El número de bytes devueltos puede ser menor o mayor que el solicitado.

**read**( [ *tamaño*] )

Lectura y devolución de bytes de *tamaño* , o si el *tamaño* no es dado o negativo, hasta EOF o si la llamada de lectura se bloquearía en el modo de no bloqueo.

**read1**( [ *tamaño*] )

Lea y devuelva hasta bytes de *tamaño* con una sola llamada en el flujo en bruto. Si al menos un byte está en búfer, solo se devuelven los bytes en búfer. De lo contrario, se realiza una llamada de lectura de flujo sin formato.

*Cambiado en vers io n 3.7:* El *tamaño del* argumento se opta ahora por io nal.

*class*( *raw* , *buffer\_size = DEFAULT\_BUFFER\_SIZE*) io.**BufferedWriter**

Un búfer que proporciona acceso de alto nivel a un objeto secuencial y grabable . Se hereda . Al escribir en este objeto, los datos normalmente se colocan en un búfer interno. El búfer se escribirá en el objeto subyacente bajo diversas condiciones, que incluyen:[RawIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.RawIOBase)[BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase)[RawIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.RawIOBase)

* cuando el búfer se vuelve demasiado pequeño para todos los datos pendientes;
* cuando [flush()](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BufferedWriter.flush" \o "io.BufferedWriter.flush)se llama;
* cuando seek()se solicita a (para [BufferedRandom](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BufferedRandom" \o "io.BufferedRandom)objetos);
* cuando el [BufferedWriter](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BufferedWriter" \o "io.BufferedWriter)objeto está cerrado o destruido.

El constructor crea una secuencia *sin*[BufferedWriter](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedWriter) formato grabable dada . Si no se proporciona el *buffer\_size* , el valor predeterminado es .[DEFAULT\_BUFFER\_SIZE](https://docs.python.org/3/library/io.html?highlight=io#io.DEFAULT_BUFFER_SIZE)

[BufferedWriter](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedWriter)proporciona o anula estos métodos en Addit io n a los de y :[BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BufferedIOBase" \o "io.BufferedIOBase)[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)

**flush**( )

Forzar los bytes retenidos en el búfer en el flujo sin procesar A se debe levantar si la secuencia en bruto bloquea.[BlockingIOError](https://docs.python.org/3/library/exceptions.html#BlockingIOError)

**write**( *b*)

Escriba el [objeto tipo bytes](https://docs.python.org/3/glossary.html#term-bytes-like-object) , *b* , y devuelva el número de bytes escritos. Cuando está en modo no bloqueante, a se eleva si el búfer necesita escribirse pero la secuencia sin procesar se bloquea.[BlockingIOError](https://docs.python.org/3/library/exceptions.html#BlockingIOError)

*class*( *raw* , *buffer\_size = DEFAULT\_BUFFER\_SIZE*) io.**BufferedRandom**

Una interfaz en búfer para flujos de acceso aleatorio. Se hereda [BufferedReader](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BufferedReader" \o "io.BufferedReader)y [BufferedWriter](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BufferedWriter" \o "io.BufferedWriter), y otros soportes seek()y tell()func io nalidad.

El constructor crea un lector y un escritor para una secuencia sin formato que se puede buscar, que aparece en el primer argumento. Si se omite el *buffer\_size,* el *valor* predeterminado es [DEFAULT\_BUFFER\_SIZE](https://docs.python.org/3/library/io.html?highlight=io#io.DEFAULT_BUFFER_SIZE).

[BufferedRandom](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedRandom)Es capaz de cualquier cosa [BufferedReader](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BufferedReader" \o "io.BufferedReader)o [BufferedWriter](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BufferedWriter" \o "io.BufferedWriter)puede hacer.

*clase*( *lector* , *escritor* , *buffer\_size = DEFAULT\_BUFFER\_SIZE*) io.**BufferedRWPair**

Un objeto / O tamponada I combinando dos Unidirect io nales objetos - uno legible, el otro grabable - en una sola bidirect io punto final nal. Se hereda .[RawIOBase](https://docs.python.org/3/library/io.html?highlight=io" \l "io.RawIOBase" \o "io.RawIOBase)[BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase)

*lector* y *escritor* son objetos que se pueden leer y escribir respectivamente. Si se omite el *buffer\_size,* el *valor* predeterminado es .[RawIOBase](https://docs.python.org/3/library/io.html?highlight=io" \l "io.RawIOBase" \o "io.RawIOBase)[DEFAULT\_BUFFER\_SIZE](https://docs.python.org/3/library/io.html?highlight=io#io.DEFAULT_BUFFER_SIZE)

[BufferedRWPair](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedRWPair)implementa todos los métodos de 's, excepto para , lo que plantea.[BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase)[detach()](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase.detach)[UnsupportedOperation](https://docs.python.org/3/library/io.html?highlight=io#io.UnsupportedOperation)

**Advertencia**

[BufferedRWPair](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedRWPair)no intenta sincronizar los accesos a sus flujos sin procesar subyacentes.No debes pasarle el mismo objeto que lector y escritor; utilizar [BufferedRandom](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BufferedRandom" \o "io.BufferedRandom)en su lugar.

Texto I / O

*clase*io.**TextIOBase**

Clase base para flujos de texto. Esta clase proporciona una interfaz basada en caracteres y líneas para transmitir I / O. No hay readinto()método porque las cadenas de caracteres de Python son inmutables. Se hereda . No hay constructor público.[IOBase](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase)

[TextIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.TextIOBase)proporciona o anula estos atributos de datos y métodos en Addit io n a los de [IOBase](https://docs.python.org/3/library/io.html?highlight=io" \l "io.IOBase" \o "io.IOBase):

**encoding**

El nombre de la codificación utilizada para decodificar los bytes de la secuencia en cadenas y para codificar las cadenas en bytes.

**errors**

La configuración de error del decodificador o codificador.

**newlines**

Una cadena, una tupla de cadenas o None, que indica las nuevas líneas traducidas hasta el momento. Dependiendo de la implantar mecanismos io n y las banderas iniciales del constructor, esto no puede estar disponible.

**buffer**

El búfer subyacente (una instancia) que trata. Esto no es parte de la API, y no puede existir en algunos implantar mecanismos io ns.[BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedIOBase)[TextIOBaseTextIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.TextIOBase)

**detach**( )

Separe el búfer binario subyacente del y devuélvalo.[TextIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.TextIOBase)

Después de que el búfer subyacente se haya desconectado, se encuentra en un estado inutilizable.[TextIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.TextIOBase)

Algunos implantar mecanismos io ns, como , no puede tener el concepto de un tampón subyacente y llamar a este método aumentará .[TextIOBase](https://docs.python.org/3/library/io.html?highlight=io" \l "io.TextIOBase" \o "io.TextIOBase)[StringIO](https://docs.python.org/3/library/io.html?highlight=io#io.StringIO)[UnsupportedOperation](https://docs.python.org/3/library/io.html?highlight=io#io.UnsupportedOperation)

*Nuevo en vers io n 3.1.*

**read**( *tamaño = -1*)

Lea y devuelva a la mayoría de los caracteres de *tamaño* del flujo como un solo [str](https://docs.python.org/3/library/stdtypes.html" \l "str" \o "str). Si el *tamaño* es negativo o None, lee hasta EOF.

**readline**( *tamaño = -1*)

Lea hasta nueva línea o EOF y devuelva una sola str. Si la transmisión ya está en EOF, se devuelve una cadena vacía.

Si se especifica el *tamaño* , como máximo se leerán los caracteres del *tamaño* .

**seek**( *Desplazamiento*[ , *de donde*] )

Cambiar la corriente postulan io n a la dada *desplazamiento* . Behav io ur depende de la *de dónde*parámetro. El valor predeterminado para *whence* es SEEK\_SET.

* SEEK\_SETo 0: buscar desde el inicio de la transmisión (el valor predeterminado); *offset* debe ser un número devuelto por , o cero. Cualquier otro *desplazamiento* valor indefinido produce behav iour.[TextIOBase.tell()](https://docs.python.org/3/library/io.html?highlight=io#io.TextIOBase.tell)
* SEEK\_CURo 1: “buscar” a la actual postulan io n; *el desplazamiento* debe ser cero, que no es una operación (todos los demás valores no son compatibles).
* SEEK\_ENDo 2: buscar hasta el final de la corriente; *el desplazamiento* debe ser cero (todos los demás valores no son compatibles).

Retorno de la nueva absoluta posit io n como un número opaco.

*Nuevo en vers io n 3.1:* Las SEEK\_\*constantes.

**tell**( )

Devolver la corriente de flujo postulan io n como un número opaco. El número no suele representar un número de bytes en el almacenamiento binario subyacente.

**write**( *s*)

Escriba la cadena *s* en la secuencia y devuelva el número de caracteres escritos.

*clase*( *búfer* , *codificación = Ninguno* , *errores = Ninguno* , *nueva línea = Ninguna* , *line\_buffering = False* , *write\_through = False*) io.**TextIOWrapper**

Un flujo de texto en búfer sobre un flujo binario. Se hereda .[BufferedIOBase](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BufferedIOBase" \o "io.BufferedIOBase)[TextIOBase](https://docs.python.org/3/library/io.html?highlight=io#io.TextIOBase)

*La codificación* proporciona el nombre de la codificación con la que se descodificará o codificará el flujo. El valor predeterminado es [locale.getpreferredencoding(False)](https://docs.python.org/3/library/locale.html" \l "locale.getpreferredencoding" \o "locale.getpreferredencoding).

*errores* es una opción io secuencia final que especifica cómo codificar y decodificar los errores se deben manejar. Pase 'strict'para elevar una [ValueError](https://docs.python.org/3/library/exceptions.html" \l "ValueError" \o "ValueError) excepción io n si hay un error de codificación (el valor por defecto de Nonetiene el mismo efecto), o pasar 'ignore'a ignorar los errores. (Tenga en cuenta que ignorar los errores de codificación puede provocar la pérdida de datos). 'replace'Hace '?'que se inserte un marcador de reemplazo (como por ejemplo ) donde hay datos con formato incorrecto.'backslashreplace'hace que los datos mal formados sean reemplazados por una secuencia de escape de barra invertida. Al escribir, se puede usar 'xmlcharrefreplace' (reemplazar con la referencia de caracteres XML correspondiente) o 'namereplace' (reemplazar con \N{...}secuencias de escape). Cualquier otro nombre de manejo de errores que se haya registrado [codecs.register\_error()](https://docs.python.org/3/library/codecs.html" \l "codecs.register_error" \o "codecs.register_error)también es válido.

*newline* controla cómo se manejan los finales de línea. Puede ser None, '', '\n', '\r', y '\r\n'. Funciona de la siguiente manera:

* Al leer la entrada de la secuencia, si es *nueva línea*None , se habilita el modo de [nuevas líneas universal](https://docs.python.org/3/glossary.html#term-universal-newlines) . Líneas en la entrada puede terminar en '\n', '\r'o '\r\n', y éstas se convierten a '\n'antes de ser devuelto a la persona que llama. Si es así '', se habilita el modo de líneas nuevas universales, pero los finales de línea se devuelven al llamante sin traducir. Si tiene alguno de los otros valores legales, las líneas de entrada solo terminan con la cadena dada, y la línea final se devuelve al llamante sin traducir.
* Al escribir la salida a la corriente, en caso *de nueva línea* es None, cualquier '\n' caracteres escritos se convierten a la línea de separación por defecto del sistema, [os.linesep](https://docs.python.org/3/library/os.html" \l "os.linesep" \o "os.linesep). Si *salto de línea* es ''o '\n', sin translat io n se lleva a cabo. Si *nueva línea* es cualquiera de los otros valores legales, cualquier '\n' carácter escrito se traduce a la cadena dada.

Si *line\_buffering* es True, flush()está implícito cuando una llamada a escribir contiene un carácter de nueva línea o un retorno de carro.

Si *write\_through* es True, write()se garantiza que las llamadas a no se almacenarán en búfer: cualquier dato escrito en el objeto se manejará inmediatamente a su *búfer* binario subyacente .[TextIOWrapper](https://docs.python.org/3/library/io.html?highlight=io" \l "io.TextIOWrapper" \o "io.TextIOWrapper)

*Cambiado en vers io n 3.3:* El *write\_through* argumento ha sido añadido.

*Cambiado en vers io n 3.3:* El valor por defecto *de codificación* está ahora locale.getpreferredencoding(False) en lugar de locale.getpreferredencoding(). No cambie temporalmente la codificación de la configuración regional utilizando [locale.setlocale()](https://docs.python.org/3/library/locale.html" \l "locale.setlocale" \o "locale.setlocale), utilice la codificación de la configuración regional actual en lugar de la codificación preferida por el usuario.

[TextIOWrapper](https://docs.python.org/3/library/io.html?highlight=io#io.TextIOWrapper)proporciona estos miembros en adic io n a las de [TextIOBase](https://docs.python.org/3/library/io.html?highlight=io" \l "io.TextIOBase" \o "io.TextIOBase)y sus padres:

**line\_buffering**

Si el búfer de línea está habilitado.

**write\_through**

Si las escrituras se pasan inmediatamente al búfer binario subyacente.

*Nuevo en vers io n 3.7.*

**reconfigure**( *\* [, codificación] [, errores] [, nueva línea] [, buffer de línea] [, write\_through]*)

Reconfigure esta secuencia de texto usando nuevas configuraciones para *codificación* , *errores* , *nueva línea* , *línea de búfer* y *escritura a través* .

Los parámetros no especificados mantienen la configuración actual, excepto que errors='strictse utiliza cuando se especifica la *codificación* pero no se especifican los *errores* .

No es posible cambiar la codificación o la nueva línea si ya se han leído algunos datos de la secuencia. Por otro lado, es posible cambiar la codificación después de la escritura.

Este método realiza un flujo de flujo implícito antes de establecer los nuevos parámetros.

*Nuevo en vers io n 3.7.*

*class*( *initial\_value = ''* , *newline = '\ n'*) io.**StringIO**

Un flujo en memoria para la E / S de texto. El búfer de texto se descarta cuando [close()](https://docs.python.org/3/library/io.html?highlight=io#io.IOBase.close)se llama almétodo.

El valor inicial del búfer se puede establecer proporcionando *initial\_value* . Si translat nueva línea io n está habilitada, las nuevas líneas serán codificados como por [write()](https://docs.python.org/3/library/io.html?highlight=io" \l "io.TextIOBase.write" \o "io.TextIOBase.write). La secuencia se coloca al inicio del búfer.

El argumento de *nueva línea* funciona como el de . El valor por defecto es considerar sólo los personajes como extremos de las líneas y de no hacer translat nueva línea io n. Si se establece *nueva línea* en , las nuevas líneas se escriben como en todas las plataformas, pero la decodificación universal de nueva línea todavía se realiza al leer.[TextIOWrapper](https://docs.python.org/3/library/io.html?highlight=io#io.TextIOWrapper)\nNone\n

[StringIO](https://docs.python.org/3/library/io.html?highlight=io#io.StringIO)proporciona este método en Addit io n a los de [TextIOBase](https://docs.python.org/3/library/io.html?highlight=io" \l "io.TextIOBase" \o "io.TextIOBase)y sus padres:

**getvalue**( )

Devuelve una strconteniendo todo el contenido del búfer. Saltos de línea se decodifican como por [read()](https://docs.python.org/3/library/io.html?highlight=io" \l "io.TextIOBase.read" \o "io.TextIOBase.read), aunque el posit corriente io no se cambia n.

Ejemplo de uso:

**import** **io**

output = io.StringIO()

output.write('First line.**\n**')

print('Second line.', file=output)

*# Retrieve file contents -- this will be*

*# 'First line.\nSecond line.\n'*

contents = output.getvalue()

*# Close object and discard memory buffer --*

*# .getvalue() will now raise an exception.*

output.close()

*clase*io.**IncrementalNewlineDecoder**

Un códec auxiliar que decodifica las líneas nuevas para [el](https://docs.python.org/3/glossary.html#term-universal-newlines) modo de [líneas nuevas universales](https://docs.python.org/3/glossary.html#term-universal-newlines) . Se hereda [codecs.IncrementalDecoder](https://docs.python.org/3/library/codecs.html" \l "codecs.IncrementalDecoder" \o "Codecs.IncrementalDecoder).

Rendimiento

Esta sección analiza el rendimiento de las implementaciones concretas de E / S proporcionadas.

I / O binario

Al leer y escribir solo grandes porciones de datos, incluso cuando el usuario solicita un solo byte, la E / S con búfer oculta cualquier ineficiencia en la llamada y la ejecución de las rutinas de E / S sin búfer del sistema operativo. La ganancia depende del sistema operativo y del tipo de E / S que se realiza. Por ejemplo, en algunos sistemas operativos modernos, como Linux, la E / S de disco sin búfer puede ser tan rápida como la E / S en búfer. Sin embargo, la conclusión es que la E / S en búfer ofrece un rendimiento predecible independientemente de la plataforma y el dispositivo de respaldo. Por lo tanto, casi siempre es preferible utilizar E / S en búfer en lugar de E / S sin búfer para datos binarios.

Texto I / O

Texto de E / S durante un almacenamiento binario (como un archivo) es significativamente más lento de lo binario de E / S en el mismo almacenamiento, ya que requiere Convers io ns entre Unicode y datos binarios usando un códec de carácter. Esto puede convertirse en algo notable al manejar grandes cantidades de datos de texto como archivos de registro grandes. Además, y son bastante lentos debido al algoritmo de reconstrucción utilizado.TextIOWrapper.tell()TextIOWrapper.seek()

[StringIO](https://docs.python.org/3/library/io.html?highlight=io#io.StringIO)Sin embargo, es un contenedor Unicode en memoria nativo y exhibirá una velocidad similar a la .[BytesIO](https://docs.python.org/3/library/io.html?highlight=io" \l "io.BytesIO" \o "io.BytesIO)

Multihilo

[FileIO](https://docs.python.org/3/library/io.html?highlight=io#io.FileIO)los objetos son seguros para subprocesos en la medida en que las llamadas al sistema operativo (como read(2)en Unix) que envuelven también son seguros para subprocesos.

Objetos almacenados temporalmente binarias (casos de [BufferedReader](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedReader), [BufferedWriter](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedWriter), [BufferedRandom](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedRandom)y [BufferedRWPair](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedRWPair)) proteger sus estructuras internas mediante una cerradura; por lo tanto, es seguro llamarlos desde múltiples hilos a la vez.

[TextIOWrapper](https://docs.python.org/3/library/io.html?highlight=io#io.TextIOWrapper) los objetos no son seguros para hilos

Reentrancia

Objetos binarios tamponadas (instancias de [BufferedReader](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedReader), [BufferedWriter](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedWriter), [BufferedRandom](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedRandom)y [BufferedRWPair](https://docs.python.org/3/library/io.html?highlight=io#io.BufferedRWPair)) no son reentrante. Mientras que las llamadas de reentrada no sucederá en situat normales ions, que pueden surgir de hacer E / S en un [signal](https://docs.python.org/3/library/signal.html" \l "module-signal" \o "señal: Establecer manejadores para eventos asíncronos.)controlador. Si un hilo intenta volver a ingresar a un objeto con búfer al que ya está accediendo, [RuntimeError](https://docs.python.org/3/library/exceptions.html" \l "RuntimeError" \o "Error de tiempo de ejecución) se genera un a. Tenga en cuenta que esto no prohíbe que un subproceso diferente entre en el objeto almacenado.

Lo anterior se extiende implícitamente a archivos de texto, ya que la [open()](https://docs.python.org/3/library/functions.html#open)func io n envolverá un objeto dentro de un buffer . Esto incluye flujos estándares y por lo tanto afecta a la incorporada en el func io n también.[TextIOWrapper](https://docs.python.org/3/library/io.html?highlight=io#io.TextIOWrapper)[print()](https://docs.python.org/3/library/functions.html#print)