PROGRAMACIÓN DE INTELIGENCIA ARTIFICIAL

PYTHON

Un programa Python se comunica con los archivos a través de las entidades abstractas llamadas **streams** o **flujos**.

```
stream = open(file, mode = 'r', encoding = None)
```

file : nombre del archivo que se asocia al stream
mode: especifica el modo de apertura del stream
encoding: codificación utilizada en archivos de texto

La primera operación que se realiza con un stream es ***open***

Modos de apertura de un stream	
r	Abre el stream en ***modo lectura.*** *El archivo debe existir y ser legible. Si no genera excepción*
w	Abre el stream en ***modo escritura.*** *Si el archivo no existe se crea. Si el archivo existe se trunca. Si no puede crearlo se genera una excepción*
a	Abre el stream en ***modo adición.*** *Si el archivo no existe se crea. Si el archivo existe se añade la información al final.*
r+	Abre el stream en modo ***lectura y actualización***: permite lectura y escritura *El archivo debe existir y debe permitir escritura. Si no, se genera una excepción*
w+	Abre el stream en modo ***escritura y actualización:*** permite lectura y escritura. *Si el archivo no existe se crea. El contenido anterior permanece intacto. Si no puede crearlo se genera una excepción*

MODO TEXTO	MODO BINARIO
rt	rb
wt	wb
at	ab
r+t	r+b
w+t	w+b

La última operación que se realiza con un stream es ***close***

```
stream.close()
```

la función close ***no espera*** *argumentos.*
El stream **no necesita** estar abierto.
Genera una excepción IOError en caso de error

```
try:
    stream = open("C:\Users\User\Desktop\file.txt", "rt")

    # El procesamiento va aquí.

    stream.close()

except Exception as exc:
    print("No se puede abrir el archivo:", exc)
```

```
from os import strerror

try:
    s = open("c:/users/user/file.txt", "rt")
    # El procesamiento va aquí.
    s.close()
except Exception as exc:
    print("Error", strerror(exc.errno))
```

La llamada a la función strerror utilizando de parámetro el atributo errno (número de error), devuelve una cadena con el significado del error.

PROGRAMACIÓN DE INTELIGENCIA ARTIFICIAL

PYTHON

Para leer el contenido de un archivo de texto se pueden utilizar diferentes métodos:

stream.read()

- ***read(number):*** lee el número de carácteres/bytes del archivo y los retorna como una cadena.
- ***read()*** es capaz de leer ***todo el archivo a la vez*** y devolverlo en ***una cadena.***
- ***readline()*** intenta leer una línea completa de texto del archivo, y la devuelve como una cadena en caso de éxito. De lo contrario, devuelve una cadena vacía.
- ***readlines()*** : sin argumentos, intenta **leer todo el contenido del archivo y devuelve una lista de cadenas, un elemento por línea del archivo**. Si se especifica un valor entero de argumento, lee tantas líneas como pueda sin sobrepasar ese número entero de bytes
  Cuando no hay nada que leer del archivo, el método devuelve una lista vacía

Para escribir en un archivo de texto se utiliza el método write():

- ***write(cadena)*** cadena es el texto que se transferirá al archivo abierto. Si se desea que el archivo se componga de varias líneas, es necesario añadir el carácter \n

PROGRAMACIÓN DE INTELIGENCIA ARTIFICIAL

PYTHON

```
counter = 0
stream = open('prim.txt', "rt")
char = stream.read(1)
while char != '':
    print(char, end='')
    counter += 1
    char = stream.read(1)
stream.close()
print("\n\nCaracteres en el archivo:", counter)
```

```
counter = 0
with open('prim.txt', "rt") as stream:
    char = stream.read(1)
    while char != '':
        print(char, end='')
        counter += 1
        char = stream.read(1)

    print("\n\nCaracteres en el archivo:", counter)
```

Empleando el ***bloque with*** no es necesario ***cerrar*** el fichero. El propio bloque se encarga de ello

Ambos fragmentos cuentan el número de caracteres en el fichero prim.txt