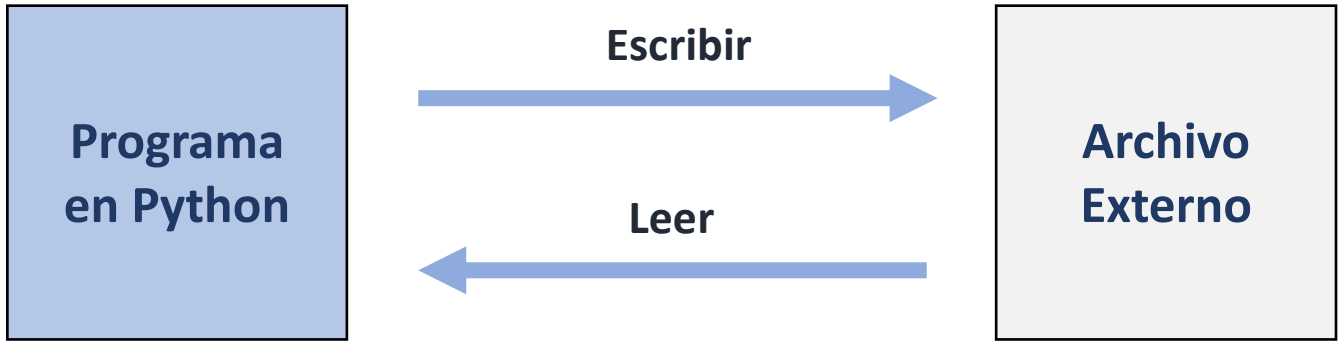
Manejo de archivos externos

La gestión de archivos externos es una herramienta que permite preprocesar, procesar y postprocesar datos de dicho archivo. Se disponen de comandos para leer y escribir (entre otras operaciones) en un documento.



Ing. Antonio Torres Sarmiento

Archivos externos

- Procesar información.
- Persistencia de datos.
- Los archivos pueden ser leídos y modificados por programas de diferentes lenguajes.
- Transferir información entre sistemas informáticos.

Ing. Antonio Torres Sarmiento

Función **open()**

- Retorna un objeto file.
- Se usa normalmente con dos argumentos posicionales y un argumento nombrado: **open(ruta_del_archivo, modo, encoding=None).**
- El primer argumento es una cadena que contiene la ruta del archivo.
- El segundo argumento describe la forma en como será usado.
  " r " : solo lectura. (Valor predeterminado)
  " w ": solo escritura.
  " r+ ": lectura y escritura
  " a ": añade información al final de forma automática.
  Otros.
- El argumento encoding permite especificar como se codificará a la hora de leer o escribir en el archivo. UTF-8 es el estándar moderno de facto, se recomienda encoding="utf-8".

Ing. Antonio Torres Sarmiento

Manejo de archivos

- Es una buena práctica usar la declaración **with** cuando manejamos objetos archivo. Tiene la ventaja de que el archivo es cerrado apropiadamente luego de que el bloque termina. Si no está utilizando la palabra clave **with**, entonces debe llamar a **archivo.close()** para cerrar el archivo y liberar inmediatamente los recursos del sistema utilizados por él.

- Cuando se lee en modo texto, por defecto se convierten los fin de lineas que son específicos a las plataformas (\n en Unix, \r\n en Windows) a solamente \n.

- Cuando se escribe en modo texto, por defecto se convierten los \n a los fin de línea específicos de la plataforma.

Ing. Antonio Torres Sarmiento

Métodos de los objetos archivos

- **read([size])**: Lee el contenido del archivo, los datos los retorna como una cadena (en modo texto) o un objeto de bytes (en modo binario). size es un argumento numérico opcional. Cuando se omite size o es negativo, el contenido entero del archivo será leído y retornado.

- **readline( )**: Lee una sola linea del archivo, el carácter de fin de línea (\n) se deja al final de la cadena, y sólo se omite en la última línea del archivo si el mismo no termina en un fin de línea.

- **readlines( )**: Retorna una lista con los valores de cada línea.

- **write(cadena)**: Escribe el contenido de la cadena al archivo, retornando la cantidad de caracteres escritos. Otros tipos de objetos necesitan ser convertidos tanto a una cadena (en modo texto) o a un objeto de bytes (en modo binario) antes de escribirlos.

Ing. Antonio Torres Sarmiento

Datos estructurados con **json**

- El módulo estándar llamado **json** puede tomar datos de Python con una jerarquía, y convertirlo a representaciones de cadena de caracteres; este proceso es llamado serialización.

  Para serializar se utiliza la función **dumps( )**.

  La función **dump()** serializa el objeto a un archivo de texto.

- Reconstruir los datos desde la representación de cadena de caracteres es llamado deserialización.

  Para decodificar un objeto nuevamente se utiliza la función **load( )**.

Ing. Antonio Torres Sarmiento