EDUCA2 – ¡Domina las Matemáticas y Programación!

Curso Intermedio de Programación en Python

Preparador: Alexanyer Naranjo

Guía Teórica N°04

Manejo de Ficheros

Hasta los momentos, todos los datos generados al crear nuestros programas son volátiles, es decir, cuando finaliza la ejecución de este, los datos generados (resultados, cálculos, entre otros) se pierden y no se almacenan en ningún sitio, pero ahora imagina que tienes la posibilidad de almacenar esos resultados en un archivo y estos persistan aún luego de que haya finalizado la ejecución de nuestro código.

Esto es lo que lograremos en este nuevo bloque del curso donde aprendemos lo relacionado al manejo de ficheros con Python y de esta manera generar archivos que almacenen los datos de nuestros códigos y que podamos acceder a estos en cualquier instante de tiempo luego de ejecutar el programa.

Para este bloque, estaremos trabajando utilizando los ficheros (archivos) aceptados por cualquier sistema operativo, los archivos de texto plano o los muy conocidos, .txt. Entonces, empecemos nuestra aventura con este tema muy interesante.

Función open(), y Métodos close(), read() y write():

Python ofrece un objeto de tipo **file** el cual nos permite la manipulación de ficheros donde podremos abrir, cerrar, escribir y leer los datos contenidos dentro de estos y así poder hacer uso de los datos obtenidos en nuestros programas:

- open(): Esta función recibe como parámetro de entrada la ubicación del fichero a través de una ruta (absoluta o relativa) y el tipo de apertura al cual queremos acceder en el fichero (hablaremos de esto más adelante).

- close(): Todo archivo que se abre, se tiene que cerrar. Esto debido a que podemos presenciar posibles errores cuando tenemos más de un programa en ejecución los cuales acceden a un mismo fichero.
- read(): Este nos permite a nosotros leer el contenido **línea a línea** de un fichero pero en caso de querer leer todo el contenido y guardarlo en una lista separada por cada línea presente, podemos utilizar **readlines().**
- write(): Este nos permite escribir dentro de un fichero el contenido que nosotros deseamos, cabe destacar que este contenido debe de ser una cadena de caracteres dado a que Python trabaja cada elemento dentro de un fichero como si fuese un string y no otro tipo de dato.

Modos de Aperturas de un fichero:

Por defecto, si a la función **open** no le enviamos ningún modo apertura, el fichero se trabajará en modo **lectura,** pero en dado caso queramos trabajar de una manera más específica, Python nos ofrece un conjunto de modos de apertura los cuales nos permitirán leer o escribir dentro de un archivo:

- Lectura (**r**): El fichero se abrirá en modo lectura donde únicamente podremos acceder al contenido de este sin posibilidad de modificarlo. En dado caso de que indiquemos la ruta de un fichero inexistente, se lanzará un error y por ello, es recomendado trabajar la apertura de fichero con un manejador de excepciones.
- Escritura (**w**): En este modo de apertura podremos modificar el contenido de un fichero según la ruta indicada, pero hay que tener mucho cuidado, dado a que, si llega a existir el fichero, el contenido que se encuentra será borrado por completo y en dado caso de que no exista, será creado en la ruta que se haya indicado como parámetro de entrada.
- Append (**a**): Este modo de apertura tiene relación directamente a **escritura,** pero a diferencia del modo **w,** en caso de que exista el fichero indicado en la ruta de entrada, el contenido no será borrado, sino que se comenzará a escribir justamente después de la última línea presente en el archivo. Y de igual modo, si no existe el fichero, este será creado en la ruta indicada.

Podemos entrar en un modo de **escritura – lectura** si indicamos justamente luego del modo el símbolo de **+** (**r+**, **w+**, **a+**) pero hay que tener cuidado con el modo **w+** dado a que, si existe el fichero, de igual manera el contenido dentro de este será borrado totalmente para así escribir un nuevo contenido.

Veamos un ejemplo donde hagamos uso del manejo de ficheros en Python.

```python
# Abrimos un archivo en modo escritura
miArchivo = open('Data.txt', 'w')

# Escribimos nuevo contenido en el archivo
for i in range(100):
  miArchivo.write('Hola\n')

# Cerramos el archivo
miArchivo.close()

# Abrimos el archivo anterior en lectura
otroArchivo = open('Data.txt')

# Leemos todo el contenido y lo imprimimos
contenido = otroArchivo.readlines()
print( contenido)

# Finalmente, cerramos el archivo
otroArchivo.close()
```

Rutas Locales y Absolutas:

Una ruta es una dirección en donde se indica el nombre de archivos o directorios (también conocidos como carpetas) separadas por barras invertidas para así acceder a un archivo en específico.

Tendremos la presencia de dos tipos de rutas, absolutas y relativas:

- Absolutas: Las **rutas absolutas** señalan la ubicación de un archivo o directorio desde el directorio raíz del sistema de archivos. Por ejemplo: **C:\Users\Alexanyer Naranjo\Desktop\Data.txt**
- Relativas: Una ruta relativa hace referencia a una ubicación que es relativa a un directorio actual. Las rutas relativas utilizan dos símbolos especiales, un punto (.) y dos puntos seguidos (..), lo que significa el directorio actual y el directorio padre. Los dos puntos seguidos se utilizan para subir en la jerarquía. Un único punto representa el directorio actual.

¡Nos vemos en la próxima clase!

De esta manera, finalizamos nuestro estudio del manejo de ficheros con Python. Cabe mencionar que también podemos trabajar con los ficheros **byte por byte,** aunque no es costumbre utilizarlo de esta manera, no está de más saber que existe la posibilidad de trabajarlo así. Ahora, tenemos la posibilidad de crear programas donde podamos hacer que los resultados persistan con el tiempo y volver a acceder a estos resultados en ejecuciones posteriores.