Introducción a la programación con Python

Iterador while

Un bucle while permite repetir la ejecución de un grupo de instrucciones mientras se cumpla una condición (es decir, mientras la condición tenga el valor True).

```
while condicion:
    cuerpo del bucle
```

Iterador while

Python evalúa la condición:

- si el resultado es True se ejecuta el cuerpo del bucle. Una vez ejecutado el cuerpo del bucle, se repite el proceso (se evalúa de nuevo la condición y, si es cierta, se ejecuta de nuevo el cuerpo del bucle) una y otra vez mientras la condición sea cierta.
- si el resultado es False, el cuerpo del bucle no se ejecuta y continúa la ejecución del resto del programa.

La variable o las variables que aparezcan en la condición se suelen llamar variables de control. Las variables de control deben definirse antes del bucle while y modificarse en el bucle while.

Iterador while

```
#El bucle se seguirá repitiendo hasta que contador llegue al valor 5

contador=0

while contador < 5:
    print (f"El contador vale {contador}")
    contador = contador + 1 #el contador aumenta su valor en 1

print("Aquí acaba el programa")
```

```
El contador vale 0
El contador vale 1
El contador vale 2
El contador vale 3
El contador vale 4
Aquí acaba el programa
```

Iterador while (Bucles infinitos)

Un bucle infinito se produce si la condición se cumple siempre.

A veces es necesario el uso de los bucles infinitos, pero normalmente se deben a errores en la lógica de programación y se perdería el control del programa.

Para detener la ejecución de uno de estos bucles se puede usar la combinacion de Ctrl+C.

Iterador while (Bucles infinitos)

```
#El bucle se seguirá repitiendo hasta el infinito

contador=0

while contador < 5:
    print (f"El contador vale {contador}")
    contador = contador + 0 #el contador no varía su valor

print("Aquí acaba el programa")
```

Ficheros

El uso de ficheros en cualquier lenguaje de programación es muy importante, dado que nos permite guardar y acceder a grandes cantidades de datos que podrían usarse para diversas tareas. (Big data)

Abrir ficheros

Para acceder a un archivo podemos utilizar la función open() y guardar su retorno en una variable:

```
fichero = open('test.txt')
```

```
<_io.TextIOWrapper name='test.txt' mode='r' encoding='cp1252'>
```

Abrir ficheros: modo de apertura

Cuando abrimos un fichero podemos hacerlo con diferentes intenciones (leer, modificar, añadir…).

- ‘r’: Por defecto, para leer el fichero.
- ‘w’: Para escribir en el fichero.
- ‘x’: Para la creación, fallando si ya existe.
- ‘a’: Para añadir contenido a un fichero existente.
- ‘b’: Para abrir en modo binario.

Por ejemplo, para abrir el fichero en modo escritura:

```
fichero = open(‘test.txt’, ‘w’)
```

Leer ficheros

Tenemos varias opciones para leer ficheros:

read(): devuelve todo el fichero.

readline(): cada vez que se llama, devuelve una línea.

readlines(): devuelve una lista donde cada elemento es una de las líneas del fichero.

Para usar estas funciones necesitaremos la variable donde abrimos el fichero, de esta forma:

```
fichero.read()
```

Cerrar ficheros

Una buena práctica es cerrar los ficheros cuando acabamos de usarlos, para evitar comportamientos extraños.

```
fichero.close()
```

Escribir en ficheros

Para escribir sobre un fichero existente o crear uno nuevo, debemos utilizar los modos de apertura que sirven para ello:

- ‘w’: Borra el fichero si ya existiese y crea uno nuevo con el nombre indicado.
- ‘a’: Añadirá el contenido al final del fichero si ya existiese
- ‘x’: Si ya existe el fichero se devuelve un error.

Escribir en ficheros

Una vez abierto el fichero en uno de los modos necesarios, contamos con 2 funciones para escribir sobre el fichero:

- write(): escribe en el fichero el contenido del argumento
- writelines(): admite una lista como argumento y escribe su contenido.

Estos métodos no añaden el salto de línea al texto, por lo que si es necesario deberíamos hacerlo nosotros. Se puede hacer añadiendo un \n al final de cada línea.

Ficheros csv (hojas de cálculo)

También podemos leer de forma sencilla este tipo de ficheros.

Utilizaremos la librería “csv”. (import csv)

Podemos abrir el fichero de la misma manera que anteriormente, con open()

Ficheros csv (hojas de cálculo)

Podemos leerlo de 2 formas:

- csv.reader(f, delimiter=' , ') -> De esta forma cada línea será una lista, donde podremos acceder a cada columna con su índice numérico

  linea[índice]

- csv.DictReader(f, delimiter=' , ') -> Así obtendremos una colección de diccionarios con la que podremos acceder a cada valor con el nombre de la columna

  linea [“nombre de columna”]