# Taller Integrador - Python.

### Clase 11

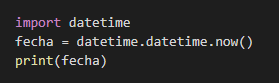
### Sábado 06/06/2020 (Chivilcoy)

### Martes 09/06/2020 CABA.

### Módulo DATETIME.

Este módulo nos facilita el trabajo con fechas en Python.

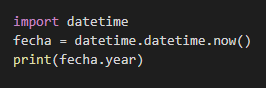
Lo primero que tenemos que hacer es importarlo.



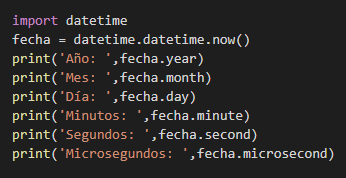
Veremos por pantalla la fecha y hora actual.

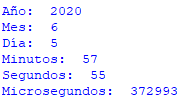
2020-06-05 18:32:15.896770

Podemos acceder al año utilizando el método YEAR de nuestro objeto fecha.



Veremos por pantalla el año actual y de la misma manera, apelando a los métodos de nuestro objeto, podemos ver los diferentes atributos de la fecha.



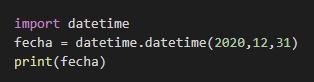


Esto vemos por pantalla:



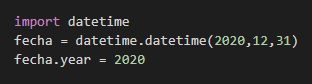
Vamos ahora a crear una fecha manualmente, que bien podría ser una fecha ingresada por teclado u obtenida desde un archivo o base de datos.

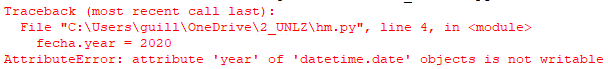
Al crear nuestro objeto fecha pasando estos atributos, podremos hacer uso de los métodos de la clase.



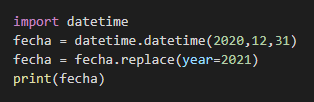


Si prestamos atención, vemos que los atributos del objeto se cargan como una tupla, por lo tanto, una vez asignados, no podremos modificarlos.



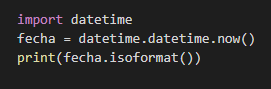


Pero si necesitamos por algún motivo modificar alguno de los atributos del objeto, podemos hacerlo mediante el método REPLACE de la siguiente manera:





Podemos utilizar diferentes formas para el formateo de la fecha, por ejemplo, el formato ISO.

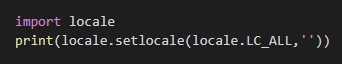




### LOCALE.

Un detalle importante a tener en cuenta a la hora de trabajar en Python con fechas, es el idioma en que está configurado, para no complicar la experiencia a los usuarios.

Vamos a importar el módulo LOCALE y ver como tenemos configurado nuestro Python.

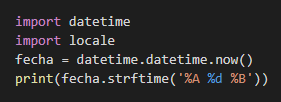




En nuestro caso, lo tenemos bien configurado en nuestro Windows y esto es importante, porque dentro de las posibilidades que nos ofrece el módulo de fechas, es mostrar días y meses por su nombre, por lo que sería conveniente poder hacerlo en el idioma del usuario final.

Si quisiéramos cambiar el idioma, por ejemplo, porque estuviese en inglés, solo debemos poner entre las comillas, las letras ‘es’.

Veamos un ejemplo de la utilidad de configurar el idioma y por qué debemos hacerlo para minimizar los resultados indeseados.

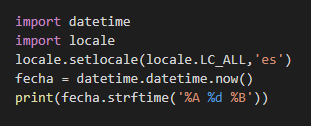




El método STRFTIME nos da información de la fecha con diferentes formatos, por ejemplo, %A simboliza el nombre del día, %d, el número de día y %B el mes.

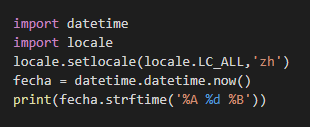
Al no configurar el idioma en ES el resultado que vemos por pantalla no es el esperado.

Hagamos el cambio de lenguaje y volvamos a ejecutar.





Si quisiéramos por ejemplo mostrar nuestros resultados en chino, usaríamos el código zh

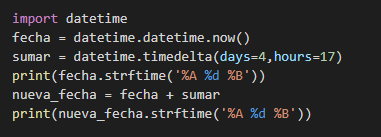




### DELTA

Veamos como sumarle una determinada cantidad de tiempo a nuestro objeto fecha.

Lo primero es crear un objeto del tipo TIMEDELTA en el que dejaremos especificados los valores que serán sumados a nuestra fecha, por ejemplo, queremos crear un delta que le sume a nuestra fecha (y a todas las fechas que necesitemos) 4 días y 17 horas.





De la misma manera, podemos restar ese delta de tiempo en lugar de sumarlo.

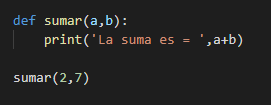
También se pueden configurar diferentes zonas horarias mediante la importación externa de un módulo llamado pyzone.

### DOCUMENTACION.

En Python todos los objetos cuentan con una variable especial llamada DOC que nos permite describir para que sirven y como se utilizan los objetos.

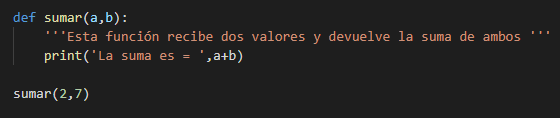
Estas variables reciben el nombre de DOCSTRNGS o cadenas de documentación y por defecto no tienen ningún valor.

Veamos por ejemplo como documentar una función que recibe dos valores y devuelve la suma de ambos.

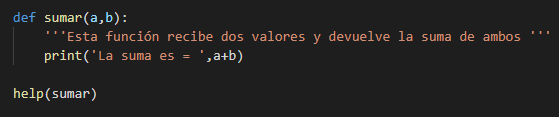




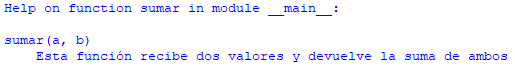
Python nos permite de manera muy sencilla establecer el valor de la DOCSTRING, lo que debemos hacer es insertar en la primera línea, después de la declaración de la variable, un comentario de triple comilla:



Si ejecutamos el código, no veremos ningún cambio, lo que debemos hacer para utilizar la documentación ingresada es hacer uso de la función HELP.

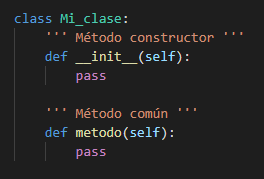


Y esto vemos por pantalla:

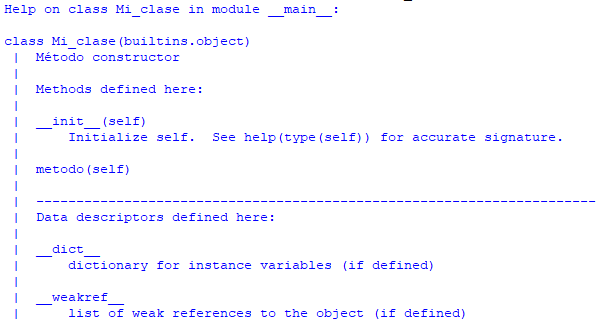


Esto mismo podemos hacer con las clases, documentando la clase en sí y cada uno de sus métodos.

Por ejemplo, tenemos la siguiente clase y queremos asignarle comentarios a la clase y sus métodos:



Ya sea que invoquemos el HELP para la clase o para un objeto creado a partir de dicha clase, obtendremos por consola los siguientes detalles:



Esto lo podemos hacer directamente desde la consola, ya sea para funciones, clases, objetos y también para módulos y paquetes.

### ERRORES.

Python, como la mayoría de los lenguajes, nos posibilita la gestión de errores en tiempo de ejecución. Los errores más comunes son los de sintaxis, que no son tan complicados porque los editores o los entornos de desarrollo nos avisan que los estamos cometiendo o en el peor de los casos, el programa deja de funcionar y nos muestra el correspondiente mensaje

de error.

Existen, sin embargo, otro tipo de errores que no se detectan tan fácilmente y que pueden pasar desapercibidos durante la etapa de desarrollo y suelen aparecer luego, porque están más ligados al uso del programa.

Este tipo errores, son los errores semánticos.

Por ejemplo, cuando tratamos de sacar con pop un elemento de una lista vacía.

Por ejemplo, cuando pedimos un valor por teclado y queremos hacer una operación matemática sin convertirlo a numero

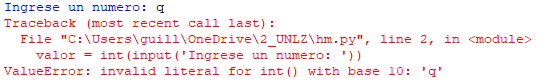
Podemos, mediante código tratar de anticiparnos a todas las posibles situaciones que se nos puedan presentar y que ocasionen que nuestro programa se detenga, pero es una tarea más que difícil en grandes desarrollos y no siempre contamos dentro del equipo, con gente (o tiempo o presupuesto) para efectuar todas las pruebas necesarias antes de sacar la versión final de nuestro proyecto.

### EXCEPCIONES.

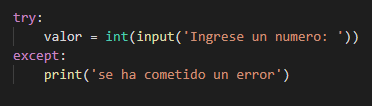
Una excepción es un bloque de código que nos permite seguir con la ejecución del programa a pesar de haberse registrado un error.

Por ejemplo, vemos que sucede cuando solicitamos el ingreso de un valor numérico para realizar una multiplicación, pero ingresamos una letra.



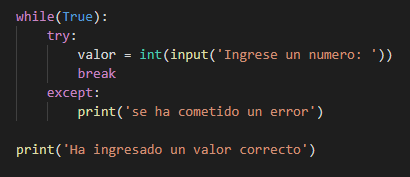


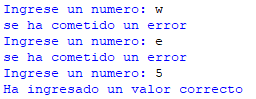
Para crear una excepción, debemos poner todo el código, que sea propenso a errores, dentro de un bloque TRY:





Si, por ejemplo, ponemos nuestro ejemplo dentro de una estructura WHILE, se seguiría ejecutando hasta que ingresemos un valor correcto.





Puedo poner un ELSE dentro si en lugar de solo salir del bloque WHILE quiero antes ejecutar algo más de código.

