Estándar de codificación y buenas prácticas de desarrollo SENA

Técnico en programación de software

Introducción

Las convenciones de código son importantes para los programadores por un gran número de razones:

- El 80% del coste del código de un programa va a su mantenimiento.
- Casi ningún software lo mantiene toda su vida el autor original.
- Las convenciones de código mejoran la lectura del software, permitiendo entender código nuevo mucho más rápidamente y más a fondo.
- Si distribuyes tu código fuente como un producto, necesitas asegurarte de que está bien hecho y presentado como cualquier otro producto.
- Para que funcionen las convenciones, cada persona que escribe software debe seguir la convención.

Es importante recalcar que las variables, métodos, archivos y demás código fuente deberá ser escrito en **inglés**, salvo que haya alguna excepción que obligue a usar una palabra en otro idioma.

Nombre de archivos

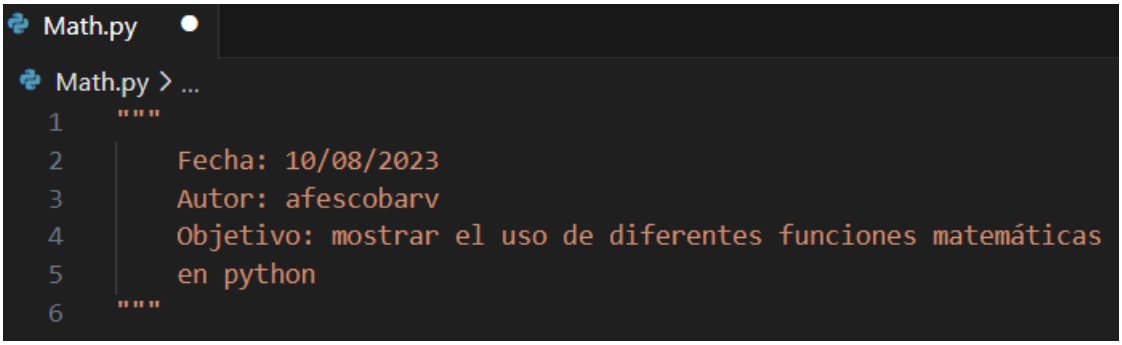
Cada fichero fuente Python contiene una única clase o interface pública, el nombre del archivo debe ser el mismo de la clase. Tanto el nombre del archivo como de la clase deben iniciar con mayúscula y deben ir en inglés. Ej.: Person, Exercise1, etc.

Los ficheros fuentes Python tienen la siguiente ordenación:

1. Comentarios de comienzo
2. Sentencias import y from
3. Declaraciones de clases e interfaces

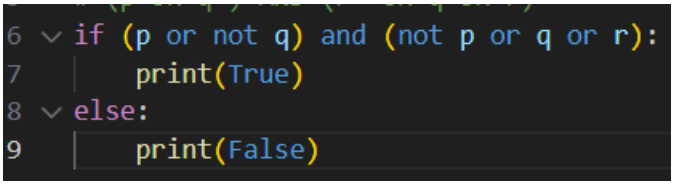
Comentarios de inicio

Todos los ficheros fuente deben comenzar con un comentario en el que se lista la fecha, autor y objetivo de la clase. Se deben usar los comentarios de bloque de python, Ejemplo:



Indentación

Se debe emplear al interior de cada bloque de función, condicional o ciclo con un tabulador como Indentación. Ejemplo:



Longitud de la línea

Evitar líneas demasiado largas, no se limita el número de caracteres, pero se establece que la línea no deba leerse con scroll horizontal para hacer más fácil la lectura.

Comentarios

Comentarios de bloque

Los comentarios de bloque se usan para dar descripciones de ficheros, métodos, estructuras de datos y algoritmos. Los comentarios de bloque se podrán usar al comienzo de cada fichero o antes de cada método. También se pueden usar en otros lugares, tales como el interior de los métodos. Los comentarios de bloque en el interior de una función o método deben ser indentados al mismo nivel que el código que describen. Un comentario de bloque debe ir precedido por una línea en blanco que lo separe del resto del código.

Comentarios de una línea

Pueden aparecer comentarios cortos de una única línea al nivel del código que siguen. Si un comentario no se puede escribir en una línea, debe seguir el formato de los comentarios de bloque.

Comentarios de remolque

Pueden aparecer comentarios muy pequeños en la misma línea que describen, pero deben ser movidos lo suficientemente lejos para separarlos de las sentencias. Si más de un comentario corto aparece en el mismo trozo de código, deben ser indentados con la misma profundidad. Ejemplo:

```
if pow == 2:
    return True      # caso especial
else:
    return False     # caso general
```

Declaraciones

Cantidad por línea

Se recomienda una declaración por línea, ya que facilita los comentarios. En otras palabras, se prefiere:

level # nivel de indentación
size   # tamaño de la tabla

Antes que:

level, size

No poner diferentes tipos en la misma línea. Ejemplo:

foo, fooarray[]; #ERROR!

Inicialización

Intentar inicializar las variables locales donde se declaran. La única razón para no inicializar una variable donde se declara es si el valor inicial depende de algunos cálculos que deben ocurrir.

Sentencias

Cada línea debe contener como mucho una sentencia. Ejemplo:

argv++ # Correcto
argc-- # Correcto
argv++ argc-- # EVITAR!

Sentencias if, if-else, if elif else

Las sentencias if si constan de más de una condición puede separarse por paréntesis para facilitar su lectura:

```
if (p or not q) and (not p or q or r):
    print(True)
else:
    print(False)
```

Sentencias for

Una sentencia for debe tener la siguiente forma:

```
for i in range(0, 10):
    print(i)
```

Tener en cuenta que las variables contador de los for pueden ser letras como i, j, k, etc.

Sentencias while

Una sentencia while debe tener la siguiente forma:

```
while number <= 10 :
    print(number)
```

Al igual que las sentencias if, si constan de más de una condición puede separarse por paréntesis para facilitar su lectura.

Convenciones de nombres

Las convenciones de nombres hacen los programas más entendibles haciéndolos más fácil de leer. También pueden dar información sobre la función de un identificador, por ejemplo, cuando es una constante, un paquete, o una clase, que puede ser útil para entender el código.

Tipos de Identificadores	**Reglas para nombres**	**Ejemplos**
Variables	Excepto las constantes, todas las variables empezarán con minúscula y en inglés. Las palabras internas que lo forman (si son compuestas) van separadas por guiones bajos. Los nombres de variables no deben empezar con los caracteres guión bajo "_" o signo del dólar "$", aunque ambos están permitidos por el lenguaje. Los nombres de las variables deben ser cortos, pero con significado. Los nombres de variables de un solo carácter se deben evitar, excepto para variables índices temporales. Nombres comunes para variables temporales son i, j, k, m, y n para enteros; c, d, y e para caracteres.	i; c my_size name user_password my_list_size
Constantes	Los nombres de las variables declaradas como constantes deben ir totalmente en mayúsculas separando las palabras con un subguión	MINIMUM_WIDTH = 4 MAXIMUM_WIDTH = 999

	("_") y en inglés.	

Algunas recomendaciones del "PEP 8" (Python Enhancement Proposal 8) son:

- Uso de espacios alrededor de operadores y después de comas en listas, tuplas y argumentos de función.
- Uso de mayúsculas iniciales en nombres de clases y archivos (CamelCase).
- Uso de comillas simples o dobles de manera coherente para cadenas (strings).

CONTROL DE VERSIONES			
Versión	**Realizada por**	**Fecha**	**Motivo**
1.0	Andrés Felipe Escobar V.	13/08/2023	Creación del documento
1.1	Andrés Felipe Escobar V.	05/09/2023	Se agregan sentencias for y while