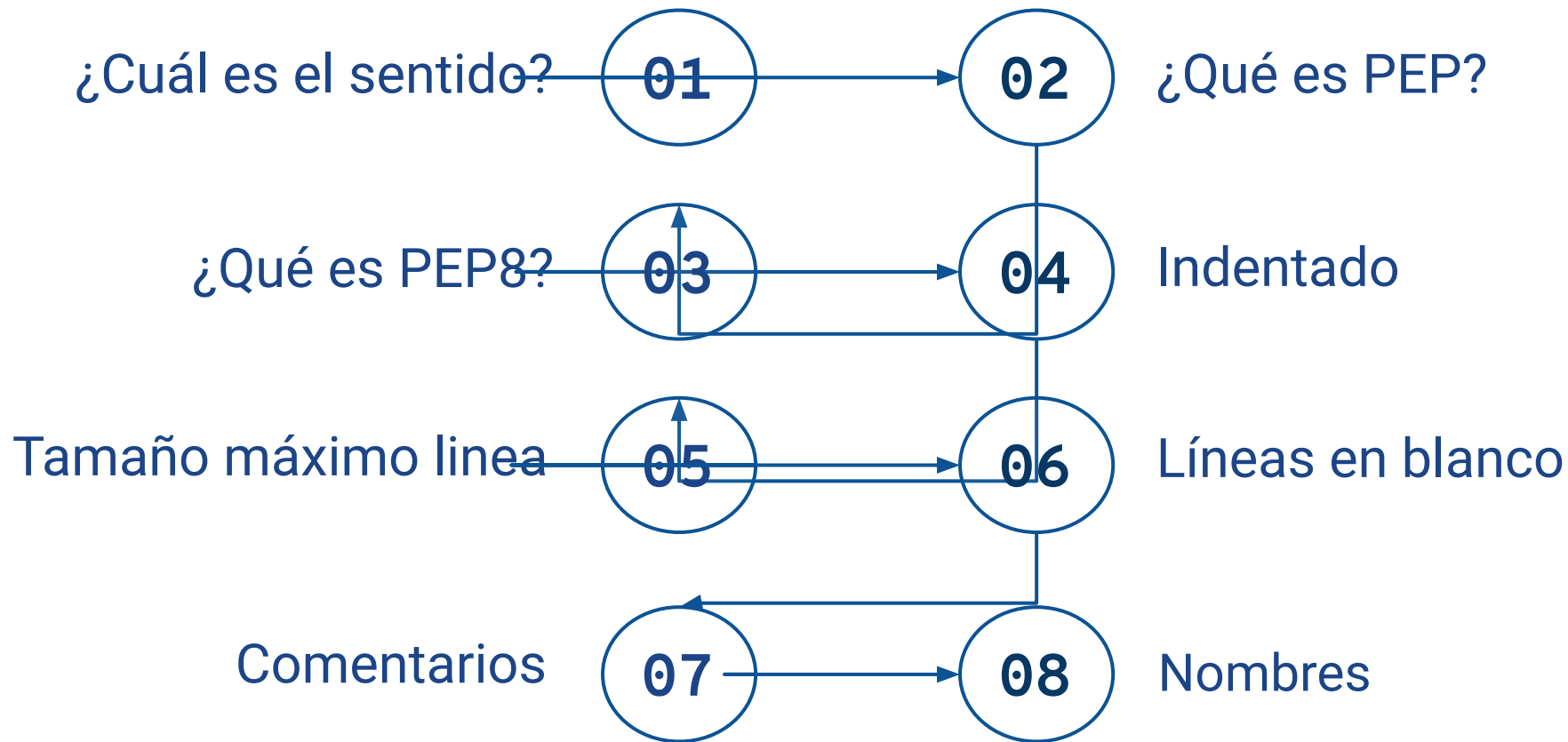


Reglas de Estilo

Programación y Laboratorio I

Reglas de estilo



¿Cuál es el sentido?

Según Guido van Rossum, el código es leído más veces que escrito, por lo que resulta importante escribir código que no sólo funcione, sino que además pueda ser leído con facilidad.

¿Qué es PEP?

Python Enhancement Proposal es un documento que proporciona información a la comunidad de Python, o que describe una nueva característica.

¿Qué es PEP8?

Es un conjunto de recomendaciones cuyo objetivo es ayudar a escribir código más legible y abarca desde cómo nombrar variables, al número máximo de caracteres que una línea debe tener.

Indentado

Python no usa `{}` para designar bloques de código como otros lenguajes de programación, sino que usa bloques identados para indicar que un determinado bloque de código pertenece a por ejemplo un `if`.

Indentado

```
if (condicion_a and  
    condicion_b):  
    print("Hola Mundo")
```

Indentado

```
def mi_funcion(primer_parametro, segundo_parametro,  
               tercer_parametro, cuarto_parametro,  
               quinto_parametro):  
    print("Hola Mundo")
```


Tamaño máximo línea

Se recomienda limitar el tamaño de cada línea a 79 caracteres, para evitar tener que hacer scroll a la derecha.

Tamaño máximo línea

```
resultado = (variable_a  
             + variable_b  
             + (variable_c - variable_d)  
             - variable_e  
             - variable_f)
```

Líneas en blanco

El uso de espacios en blanco mejora la legibilidad del código, y es por lo que la PEP8 indica dónde debemos usar espacios y dónde no.

Líneas en blanco

Correcto

```
x = 5
```

Incorrecto

```
x=5
```

Correcto

```
if x == 5:  
    pass
```

Incorrecto

```
if x==5:  
    pass
```

Correcto

```
var_a = 0  
variable_b = 10  
otra_variable_c = 3
```

Incorrecto

```
var_a          = 0  
variable_b     = 10  
otra_variable_c = 3
```

Cualquier comentario que contradiga el código es peor que ningún comentario.

Si actualizamos el código, se debe actualizar los comentarios para evitar crear inconsistencias

Evitar comentarios poco descriptivos que no aporten nada más allá de lo que ya se ve a simple vista.

Comentarios

Incorrecto

`x = x * 1.21` # Multiplica por 1.21 a la variable x

Correcto

`x = x * 1.21` # Agrega el 21% de IVA

Funciones: Uso de **snake_case**, letras en minúscula separadas por guión bajo: `mi_funcion`.

Variables: Al igual que las funciones: `variable`, `mi_variable`.

Clases: Uso de **CamelCase**, usando mayúscula y sin barra baja: `MiClase`, `ClaseDePrueba`.

Métodos: Al igual que las funciones, usar snake case: metodo, mi_metodo.

Constantes: Nombrarlas usando mayúsculas y separadas por guión bajas: UNA_CONSTANTE

Módulos: Igual que las funciones: modulo.py, mi_modulo.py.

YAPA

The background features a series of overlapping chevron shapes pointing to the right. The colors transition from a dark charcoal blue on the left to a light sky blue on the right. A prominent teal-colored chevron is layered over the darker shades in the center.

Encoding (ñ)

Los archivos se codifican por defecto en ASCII para Python 2 y UTF-8 para Python 3, por lo que será necesario definir la codificación que usemos cuando pretendamos usar otro tipo.

Encoding (ñ)

```
# -*- coding: latin-1 -*-  
print("La acentuación del Español")  
# La acentuación del Español
```