UTN AVELLANEDA
Tecnicatura Universitaria en Programación

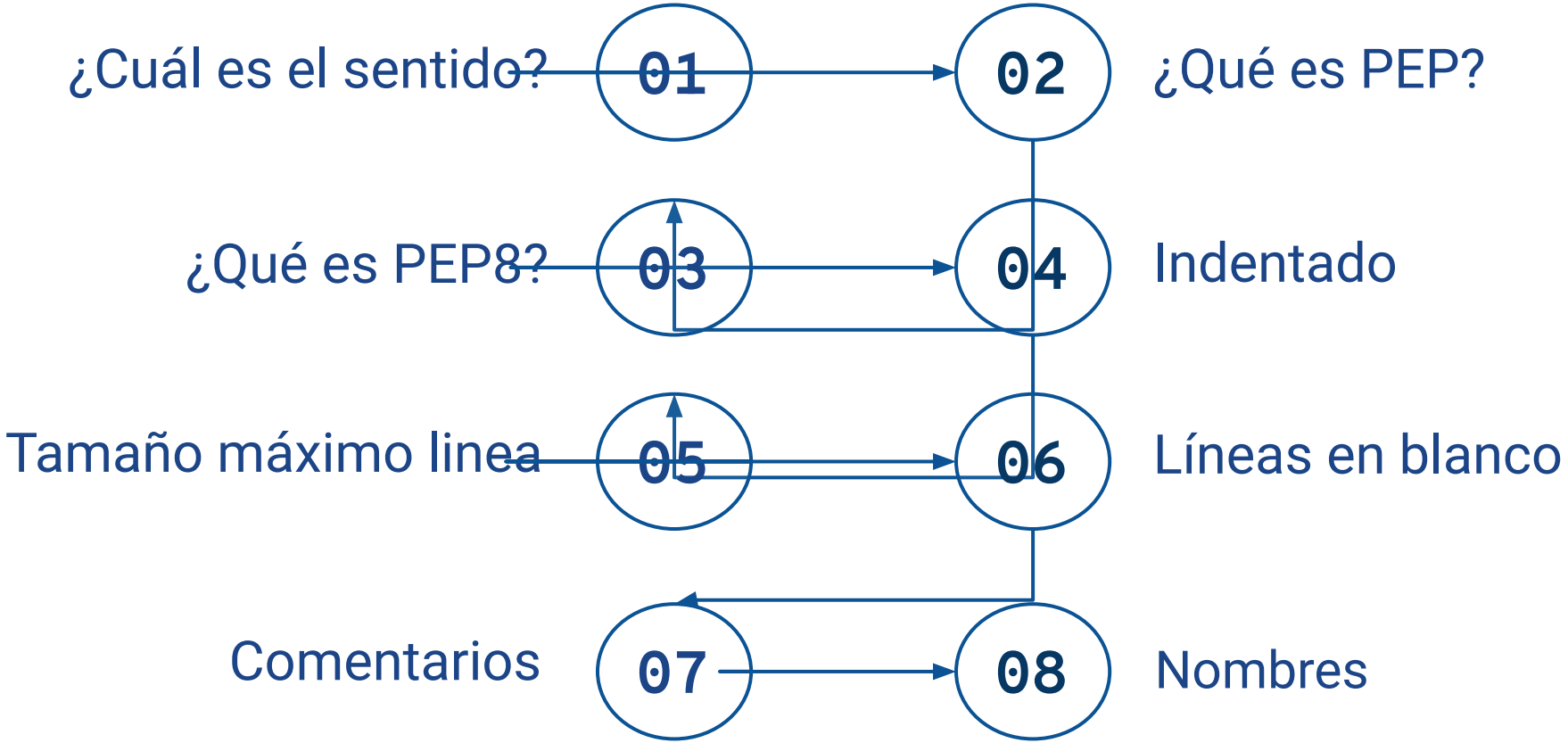
Reglas de Estilo

Programación y Laboratorio I

Versión ‘24

UTN AVELLANEDA
Tecnicatura Universitaria en Programación

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 **E**nhancement **P**roposal 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 indentados 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 linea

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

Tamaño máximo linea

```
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

```python
# Correcto
x = 5

# Incorrecto
x=5
```

```python
# Correcto
if x == 5:
    pass

# Incorrecto
if x==5:
    pass
```

```python
# Correcto
var_a = 0
variable_b = 10
otra_variable_c = 3

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

Comentarios

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

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

Comentarios

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
```

Nombres

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.

Nombres

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

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
```