

Nombres de variables

Edison Achalma

Escuela Profesional de Economía, Universidad Nacional de San Cristóbal de Huamanga

Resumen

Este abstract será actualizado una vez que se complete el contenido final del artículo.

Tabla de contenidos

Introduction	2
0.1 Motivación	2
0.2 Intuición	2
0.3 Modelo mental	2
0.3.1 Reglas obligatorias	2
0.3.2 Convenciones	2
0.4 Desarrollo	2
0.4.1 1. Reglas estrictas (lo que es legal o ilegal)	2
0.4.2 2. Convenciones de estilo para nombres compuestos	3
0.4.3 3. Palabras reservadas (no usar)	3
0.4.4 4. Convenciones especiales (por contexto)	3
0.5 Ejemplos	3
0.6 Errores comunes	4
0.7 Buenas prácticas	5
0.8 Ejercicio	6
0.9 Síntesis	6
0.10 Qué sigue	6
1 Publicaciones Similares	7

Edison Achalma  <https://orcid.org/0000-0001-6996-3364>

El autor no tiene conflictos de interés que revelar. Los roles de autor se clasificaron utilizando la taxonomía de roles de colaborador (CRediT; <https://credit.niso.org/>) de la siguiente manera: Edison Achalma: conceptualización, metodología, análisis formal, investigación, recursos, curación de datos, redacción, visualización, supervisión, administración del proyecto

La correspondencia relativa a este artículo debe dirigirse a Edison Achalma, Escuela Profesional de Economía, Universidad Nacional de San Cristóbal de Huamanga, Portal Independencia N° 57, Ayacucho, AYA 5001, Perú, Email: elmer.achalma.09@unsch.edu.pe

Nombres de variables

0.1 Motivación

Los nombres de variables son la forma en que te comunicas con otros programadores y contigo mismo. Un buen nombre hace que tu código sea **autoexplicativo**, reduciendo la necesidad de comentarios y facilitando el mantenimiento. Elegir nombres claros no es solo estilo: es una habilidad fundamental que distingue a un programador novato de uno profesional.

0.2 Intuición

Piensa en los nombres de variables como las etiquetas de los cajones en una oficina. Si cada cajón tiene una etiqueta clara (“Facturas”, “Contratos”, “Informes”), cualquiera puede encontrar lo que busca. Si usas etiquetas como “Cosa1”, “Cosa2” o “Eso”, nadie (ni tú) sabrá qué hay dentro. En programación pasa igual: `total_ventas` es mucho mejor que `x` o `temp`.

0.3 Modelo mental

Las reglas de nombres en Python son **estrictas** (el intérprete te dará error si no las cumples) y **convencionales** (buenas prácticas que no son obligatorias pero que mejoran la legibilidad).

0.3.1 Reglas obligatorias

- Solo letras (A-Z, a-z), números (0-9) y guión bajo (`_`)
- No puede empezar con número
- No puede ser una palabra reservada de Python
- Sensible a mayúsculas y minúsculas

0.3.2 Convenciones

- `snake_case` (recomendado por PEP 8)
- Nombres descriptivos
- Constantes en MAYUSCULAS
- Clases en PascalCase

0.4 Desarrollo

0.4.1 1. Reglas estrictas (lo que es legal o ilegal)

Nombres legales :

```
nombre = "Ana"           # Letras
_nombre = "Ana"          # Empieza con guión bajo
nombre2 = "Ana"           # Termina con número
nombre_completo = "Ana"  # Guión bajo
Nombre = "Ana"            # Mayúscula inicial (pero no recomendado)
NOMBRE = "Ana"            # Todo mayúsculas (para constantes)
```

Nombres ilegales :

```
2nombre = "Ana"      # Empieza con número
mi-nombre = "Ana"    # Contiene guión (-)
mi nombre = "Ana"    # Contiene espacio
if = "Ana"           # Palabra reservada
```

0.4.2 2. Convenciones de estilo para nombres compuestos

Cuando un nombre tiene varias palabras, usa una de estas convenciones:

Estilo	Ejemplo	Uso recomendado
Snake Case	mi_nombre_variable	Variables y funciones (PEP 8)
Camel Case	miNombreVariable	A veces usado en JavaScript, no en Python
Pascal Case	MiNombreVariable	Clases en Python

Recomendación: usa snake_case para variables en Python:

```
# Snake Case (recomendado)
precio_total = 100
nombre_usuario = "Carlos"
lista_productos = ["manzana", "pera"]

# Camel Case (no recomendado en Python)
precioTotal = 100
nombreUsuario = "Carlos"
```

0.4.3 3. Palabras reservadas (no usar)

Estas palabras tienen un significado especial en Python y no pueden usarse como nombres:

```
# No usar como variables
# False, class, finally, is, return, None, continue, for, lambda, try,
# True, def, from, nonlocal, while, and, del, global, not, with,
# as, elif, if, or, yield, assert, else, import, pass, break, except, in, raise
```

0.4.4 4. Convenciones especiales (por contexto)

Convención	Ejemplo	Significado
Mayúsculas	PI = 3.14159	Constante (no debe cambiar)
Guión bajo al inicio	_interno	“Privado” (por convención)
Doble guión bajo	__magico__	Métodos especiales de Python
Minúsculas y guión bajo	variable_normal	Variables estándar

0.5 Ejemplos

Mínimo — Nombres legales e ilegales:

```
# Legales
nombre = "Ana"
_edad = 25
precio_producto = 99.99
total_1 = 100

# Ilegales (descomentar da error)
# 1er_lugar = "primero"      # Empieza con número
# mi-variable = 10           # Tiene guión
# class = "programacion"     # Palabra reservada
# mi variable = "texto"      # Tiene espacio
```

Progresivo — Nombres descriptivos vs. nombres confusos:

```
# Malos nombres (difíciles de entender)
a = 12.5
b = "Juan"
c = a * 1.21
print(c)

# Buenos nombres (autoexplicativos)
precio_base = 12.5
nombre_cliente = "Juan"
precio_con_iva = precio_base * 1.21
print(precio_con_iva)
```

Real — Sistema de gestión de inventario:

```
# Configuración del sistema
LIMITE_INVENTARIO = 100      # Constante (mayúsculas)
ruta_archivo = "/data/inventario.csv" # Variable normal
_contador_interno = 0        # Variable "privada"

# Datos de productos
producto_nombre = "Laptop"
producto_precio = 850.50
producto_stock = 15
producto_activo = True

# Mensaje de estado
print("Producto:", producto_nombre)
print("Stock:", producto_stock, "unidades")
print("Stock bajo:", producto_stock < 10) # Resultado: False
```

0.6 Errores comunes

Error	Síntoma	Cómo evitarlo
Espacio en nombre	SyntaxError: invalid syntax	Usa _ en lugar de espacio
Número al inicio	SyntaxError: invalid syntax	Empieza con letra o _
Palabra reservada	SyntaxError: invalid syntax	Consulta la lista de palabras reservadas
Guión en lugar de guión bajo	SyntaxError: invalid syntax	Usa _ no -
Mayúsculas incorrectas	NameError: name 'x' is not defined	Python es sensible a mayúsculas: Variable ≠ variable

0.7 Buenas prácticas

1. Nombres descriptivos pero concisos:

```
# Demasiado corto
a = 100

# Demasiado largo
precio_total_de_la_compra_sin_iva = 100

# Justo
precio_base = 100
```

2. Usa verbos para funciones, sustantivos para variables:

```
# Función: acción (verbo)
def calcular_total():
    pass

# Variable: dato (sustantivo)
total = 100
```

3. Sé consistente:

```
# Inconsistente
nombre_usuario = "Ana"
user_email = "ana@mail.com"

# Consistente
nombre_usuario = "Ana"
email_usuario = "ana@mail.com"
```

4. Evita abreviaturas no estándar:

```
# Confuso
prv = "Programación"

# Claro
curso = "Programación"
```

5. Usa nombres que tengan sentido en el contexto del programa.

0.8 Ejercicio

Básico — Corrige nombres ilegales

Los siguientes nombres de variables tienen errores. Corrígelos para que sean válidos y sigan las convenciones de Python:

```
# Corrige estos nombres ilegales/malos
1er_trimestre = 2024
total-ventas = 15000
mi variable = "casa"
class = "Python"
NombreDeUsuario = "ana"
precioTotal = 99.99
```

Solución (no mires hasta haberlo intentado):

```
# Versión corregida (snake case)
primer_trimestre = 2024
total_ventas = 15000
mi_variable = "casa"
clase = "Python" # O mejor: nombre_clase
nombre_usuario = "ana"
precio_total = 99.99
```

0.9 Síntesis

- **Reglas obligatorias:** letras, números, _; no empezar con número; sin espacios ni -; sin palabras reservadas.
- **Sensible a mayúsculas:** nombre ≠ Nombre ≠ NOMBRE.
- **Snake Case:** convención recomendada para variables (mi_variable).
- **Descriptivos:** usa nombres que revelen el propósito de la variable.
- **Constantes en mayúsculas:** LIMITE = 100.
- **Palabras reservadas:** no usar if, for, class, while, etc.

0.10 Qué sigue

- **Relacionado:** variables (asignación, tipos), palabras reservadas
- **Siguiente tema:** Tipos de datos — aprenderás a manejar diferentes tipos de datos y sus propiedades.

1 Publicaciones Similares

Si te interesó este artículo, te recomendamos que explores otros blogs y recursos relacionados que pueden ampliar tus conocimientos. Aquí te dejo algunas sugerencias:

1.  [Instalacion De Anaconda](#)
2.  [Configurar Entorno Virtual Python Anaconda](#)
3.  [01 Introducion A La Programacion Con Python](#)
4.  [02 Variables Expresiones Y Statements Con Python](#)
5.  [03 Objetos De Python](#)
6.  [04 Ejecucion Condicional Con Python](#)
7.  [05 Iteraciones Con Python](#)
8.  [06 Funciones Con Python](#)
9.  [07 Dataframes Con Python](#)
10.  [08 Prediccion Y Metrica De Performance Con Python](#)
11.  [09 Metodos De Machine Learning Para Clasificacion Con Python](#)
12.  [10 Metodos De Machine Learning Para Regresion Con Python](#)
13.  [Visualizacion De Datos Con Python](#)

Esperamos que encuentres estas publicaciones igualmente interesantes y útiles. ¡Disfruta de la lectura!