Módulo 2.1: Datos y Variables en Python 🧱🔡

Prof. Carlos Cedeño

¡Bienvenid@ al Módulo 2.1! En los módulos anteriores, sentamos las bases conceptuales de la programación y tuvimos una introducción específica a Python, su sintaxis y las herramientas que podemos utilizar. Ahora, es momento de sumergirnos en la práctica y comenzar a escribir código Python para manipular datos.

En este módulo, aprenderás sobre:

- **Variables:** Cómo almacenar y nombrar información en tus programas.
- **Conversión de Tipos de Datos:** Cómo cambiar datos de un tipo a otro (ej. de texto a número).

1. Variables: Almacenando Información

¿Qué son las Variables?

En programación, una **variable** es como una caja con una etiqueta donde puedes guardar un valor o pieza de información para usarla más tarde en tu programa. En lugar de referirte al valor directamente cada vez, te refieres a la etiqueta de la caja (el nombre de la variable). Esto hace que los programas sean más legibles, flexibles y fáciles de mantener.

Imagina que estás calculando el área de un rectángulo. Podrías tener variables para el `ancho`, el `alto` y luego una para el `area`.

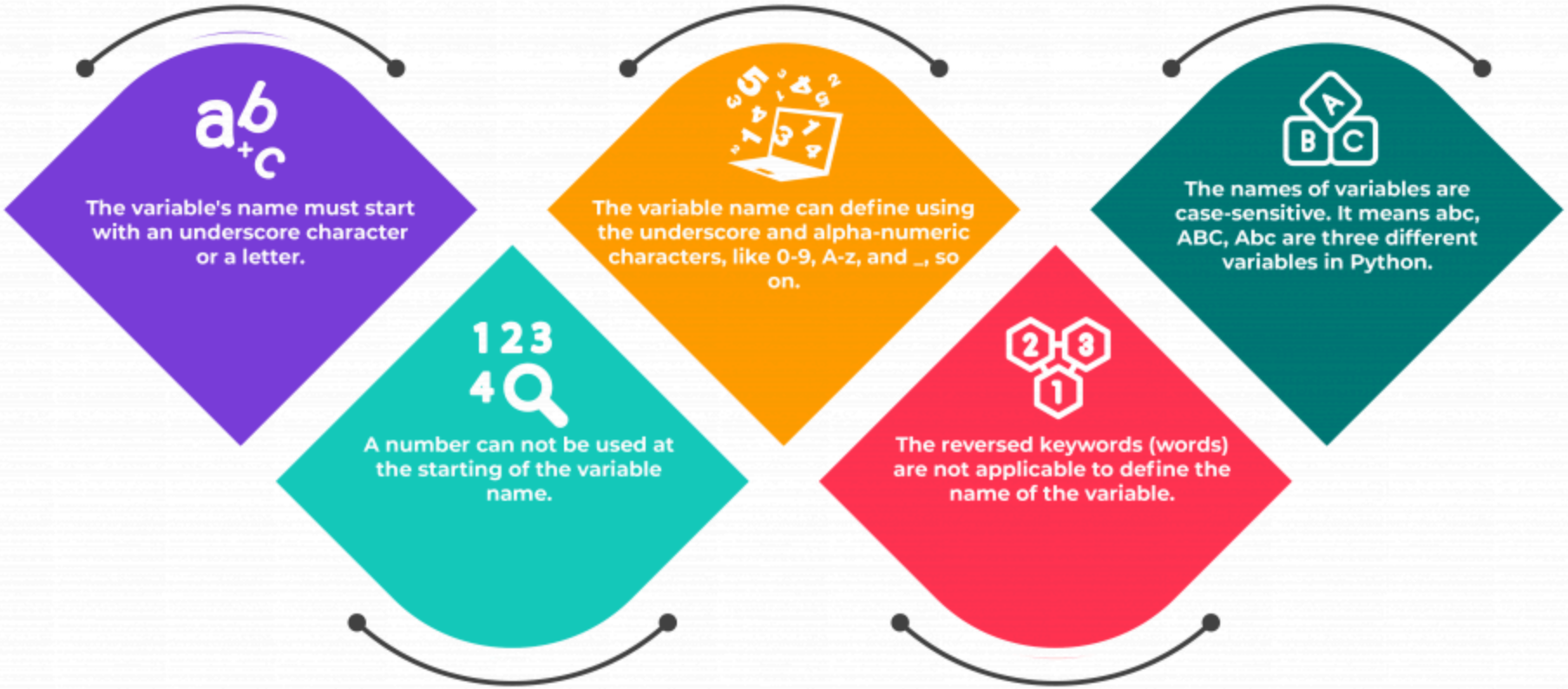
Nombrando Variables: Reglas y Buenas Prácticas

Python tiene algunas reglas y convenciones para nombrar variables:

- **Reglas (Obligatorias):**
  - Los nombres de las variables solo pueden contener letras (a-z, A-Z), números (0-9) y el guion bajo (`_`).
  - No pueden comenzar con un número. (Ej: `1variable` es incorrecto, `variable1` es correcto).
  - No pueden ser ninguna de las palabras reservadas de Python (keywords como `if` , `else` , `for` , `while` , `def` , `class` , etc., que tienen un significado especial en el lenguaje).
  - Python distingue entre mayúsculas y minúsculas (sensible a mayúsculas), así que `edad` , `Edad` y `EDAD` son tres variables diferentes.

Is there any rule for creating variables in Python?

Yes, there are certain rules that users need to follow for creating a variable in Python.
And the rules are as follows:



- **Buenas Prácticas (Convenciones Recomendadas):**
  - **Nombres Descriptivos:** Usa nombres que indiquen claramente qué información contiene la variable (ej: `nombre_cliente` en lugar de `nc` o `x`).
  - `snake_case` **para Variables y Funciones:** Es la convención más común en Python. Consiste en escribir las palabras en minúsculas separadas por guiones bajos (ej: `tasa_de_interes`, `calcular_impuesto`).
  - **Evita Nombres Demasiado Cortos o Largos:** Busca un equilibrio.
  - **Sé Consistente:** Usa el mismo estilo de nomenclatura en todo tu proyecto.

Asignación de Valores

Para guardar un valor en una variable, se utiliza el operador de asignación, que es el signo igual (`=`).

```
nombre_estudiante = "Ana Pérez"
edad = 23
altura_metros = 1.65
es_mayor_de_edad = True
```

Una variable puede ser asignada con un valor y luego reasignada con un nuevo valor (incluso de un tipo diferente, como veremos a continuación).

```
contador = 0
print(f"Valor inicial del contador: {contador}")

contador = contador + 1
print(f"Contador después de incrementar: {contador}")

contador = "Terminado"
print(f"Contador ahora es: {contador}")
```

Salida Esperada:

```
Valor inicial del contador: 0
Contador después de incrementar: 1
Contador ahora es: Terminado
```

También puedes asignar el mismo valor a múltiples variables simultáneamente:

```
x = y = z = 100
print(x)
print(y)
print(z)
```

Salida Esperada:

```
100
100
100
```

Tipado Dinámico en Python

Una característica importante de Python es que es un lenguaje de **tipado dinámico**. Esto significa que no necesitas declarar explícitamente el tipo de dato de una variable antes de usarla (como se hace en lenguajes de tipado estático como Java o C++).

Python determina automáticamente el tipo de dato de una variable en el momento en que se le asigna un valor.

```python
mi_dato = 101
print(f"Valor: {mi_dato}, Tipo: {type(mi_dato)}")
# Salida Esperada: Valor: 101, Tipo: <class 'int'>

mi_dato = "Hola, Mundo!"
print(f"Valor: {mi_dato}, Tipo: {type(mi_dato)}")
# Salida Esperada: Valor: Hola, Mundo!, Tipo: <class 'str'>

mi_dato = 3.14159
print(f"Valor: {mi_dato}, Tipo: {type(mi_dato)}")
# Salida Esperada: Valor: 3.14159, Tipo: <class 'float'>
```

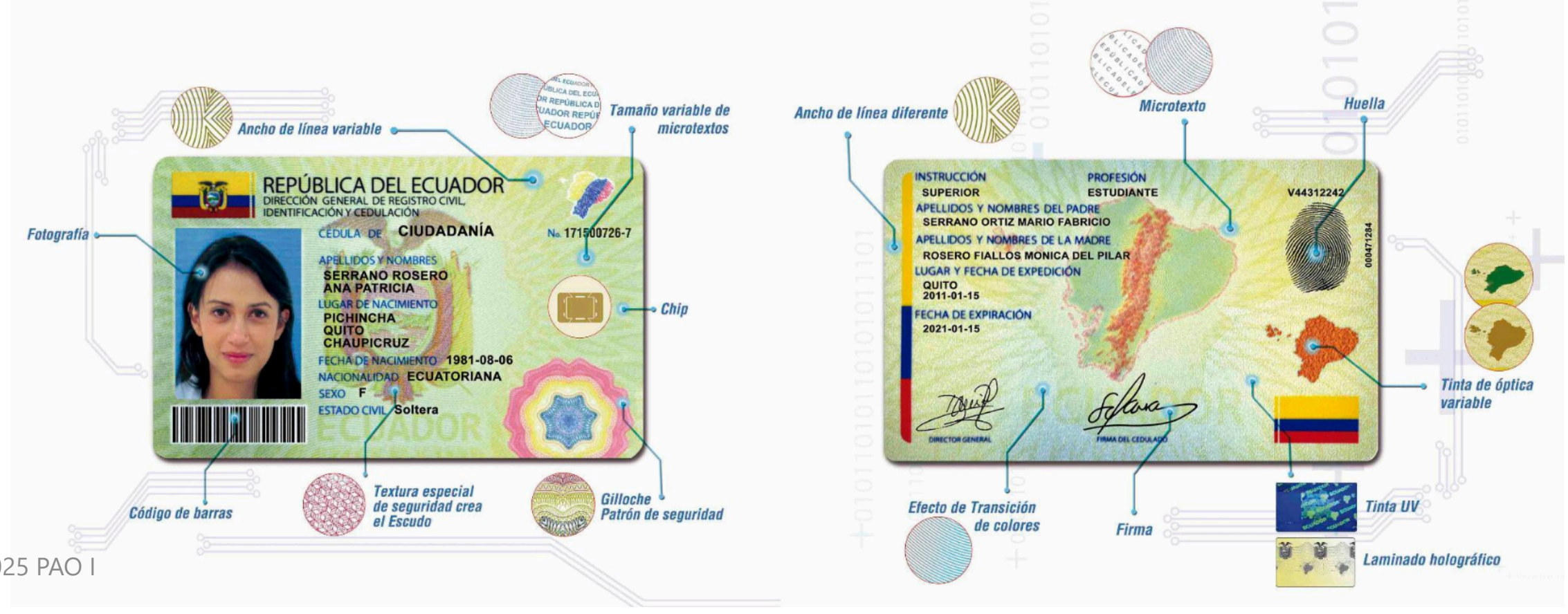
La función type() es muy útil para averiguar el tipo de dato actual de una variable.

Si bien el tipado dinámico ofrece flexibilidad, también significa que debes ser consciente del tipo de dato con el que estás trabajando, ya que ciertas operaciones solo son válidas para ciertos tipos (por ejemplo, no puedes dividir un texto).

Ejercicios Prácticos de Variables

Almacenando datos de una persona

Crear las variables y asignarle los valores



Primer Checkpoint

Hemos cubierto los aspectos fundamentales de las variables en Python:

Las variables son **nombres** que damos a **espacios de memoria** para almacenar datos.

Existen reglas y convenciones importantes para nombrar variables, siendo snake_case y nombres descriptivos las prácticas recomendadas.

El operador de asignación (=) se usa para dar valores a las variables.

Python utiliza tipado dinámico, lo que significa que el tipo de una variable se determina en tiempo de ejecución según el valor asignado.

Próxima clase: Conversión de tipos de datos