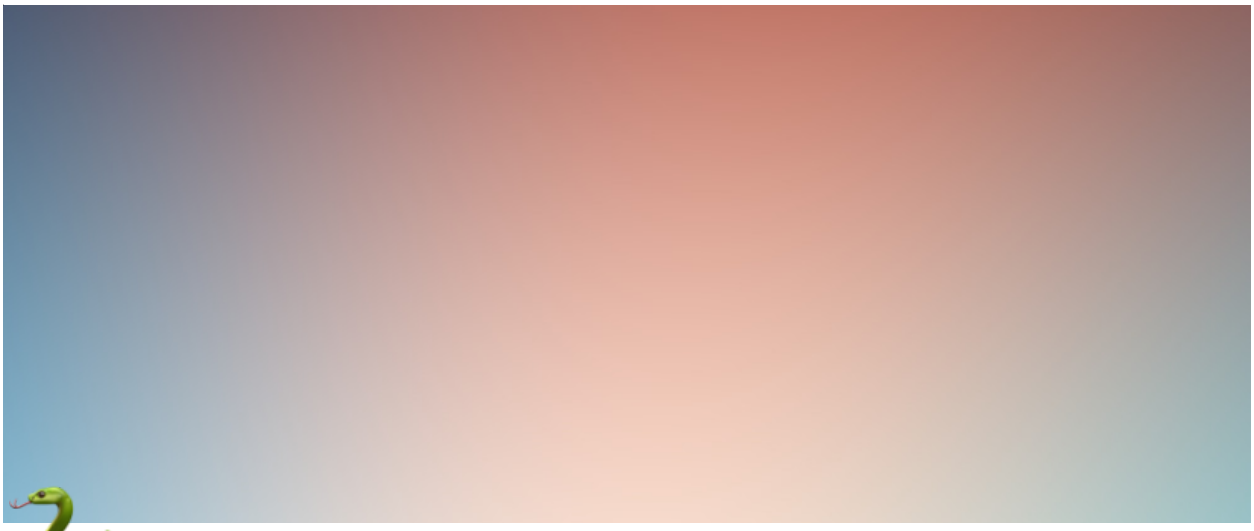

# 🖥️ Clase 7: Operaciones de Entrada y Salida en Consola

Cuando trabajamos en proyectos de programación, es común interactuar con el usuario, como en casos donde se nos pide ingresar datos como el correo o la contraseña para ejecutar alguna acción. Esta lógica es fundamental para recibir información desde la consola, como veremos en el siguiente ejemplo.

## 📝 Ejemplo Básico: Solicitar Nombre

Primero, vamos a crear una variable llamada `nombre` y le asignaremos el resultado de la función `input()` para que el usuario pueda ingresar su nombre:

```
nombre = input("Por favor, ingresa tu nombre: ")
```

Cuando ejecutamos el código, se abrirá una sección donde podemos introducir información. Supongamos que el usuario escribe "Calle Trece". Si queremos ver lo que ingresó el usuario, debemos utilizar la función `print()` para mostrar el resultado en consola:

```
print(nombre)
```

Si lo ejecutamos, y el usuario ingresa "Michael", entonces veremos "Michael" impreso en pantalla.

## 📝 Solicitar Edad

Ahora, vamos a solicitarle al usuario que ingrese su edad. Para eso, creamos otra variable llamada `edad` y usamos la misma lógica:

```
edad = input("Por favor, ingresa tu edad: ")
print(edad)
```

Cuando ejecutamos el programa, el usuario ingresará su nombre y su edad. Si por ejemplo se ingresa "29", esta también se mostrará en pantalla.

## ⚠️ Atención al Tipo de Datos

Algo importante a tener en cuenta es que **todo lo que ingresa el usuario a través de `input()` se interpreta como una cadena de texto (string)**, incluso si ingresamos un número. Veamos esto en acción consultando el tipo de dato de las variables `nombre` y `edad`:

```
print(type(nombre))  # Mostrará <class 'str'>
print(type(edad))    # También mostrará <class 'str'>
```

Si el usuario ingresa "Charly" y "29", Python nos dirá que ambos son de tipo `str`.

## 🎯 Conversión de Tipos de Datos (Casting)

Si queremos que la **edad** sea un número entero en lugar de una cadena, debemos convertirla explícitamente usando `int()`. A esto se le llama "casting". Así lo haríamos:

```
edad = int(input("Por favor, ingresa tu edad: "))
print(type(edad))  # Ahora será <class 'int'>
```

Ahora, cuando el usuario ingrese "29", Python lo interpretará como un número entero.

## 🔢 Conversión a Otros Tipos

De igual manera, podemos convertir el dato a otros tipos como **flotante** (números decimales) utilizando `float()`:

```
edad = float(input("Por favor, ingresa tu edad: "))
print(type(edad))  # Mostrará <class 'float'>
```

Si el usuario ingresa "31.5", el resultado será un número decimal de tipo `float`.

## 🚨 Manejo de Errores

Si intentamos convertir una cadena no numérica a un tipo de dato numérico, Python lanzará un error del tipo `ValueError`. Por ejemplo, si se espera un número pero el usuario ingresa un nombre, obtendremos un error:

```
edad = int(input("Ingresa tu edad: "))  # Si el usuario ingre
sa texto en lugar de números, generará un error
```

Para evitar que el programa se detenga, podemos usar estructuras de control como `try` y `except` para manejar los errores:

```
try:
    edad = int(input("Por favor, ingresa tu edad: "))
except ValueError:
    print("❌ Error: Debes ingresar un número.")
```

---

Con esto, hemos aprendido a **recibir datos** del usuario en la consola y cómo convertirlos al tipo de dato adecuado para evitar errores. 🤷