SESIÓN SENTENCIAS BÁSICAS Y VARIABLES EN PYTHON

CONTENIDOS:

- Variables.
- Tipos de dato fundamentales: entero, decimal, cadena de caracteres, booleano.
- Expresiones aritméticas.
- Conversiones de tipo.
- Impresión y entrada de datos en consola.
- Creación y ejecución de un script Python.

VARIABLES

Una **variable** es un espacio en la memoria de la computadora donde podemos almacenar información. En Python, las variables no necesitan ser declaradas previamente con un tipo específico; simplemente se asigna un valor a un nombre de variable, y Python se encarga de gestionar su tipo de dato automáticamente. Para crear una variable en Python, solo necesitamos asignarle un valor:

```
x = 5
nombre = "Juan"
```

Ilustración 1 Ejemplo de variables

En este ejemplo, **x** es una variable que almacena el número entero 5, y **nombre** es una variable que almacena una cadena de texto "Juan". En Python, no es necesario especificar el tipo de la variable; Python lo deduce de acuerdo con el valor que se asigna.

Buenas prácticas al trabajar con variables:

- **Nombres descriptivos**: Los nombres de las variables deben ser fáciles de entender y reflejar su propósito. Por ejemplo, en lugar de usar a, es mejor usar edad si la variable almacena una edad.

- **Evitar nombres reservados**: Algunos nombres son palabras clave en Python, como if, else, for, while, entre otros, y no deben usarse como nombres de variables.

TIPOS DE DATO FUNDAMENTALES EN PYTHON

En Python, los datos que podemos almacenar en las variables son de diferentes tipos. Los más comunes son:

1. **Entero (int)**: Representa números enteros, es decir, sin decimales.

```
edad = 25
```

Ilustración 2 Ejemplo de dato Entero (int)

2. **Decimal (float)**: Representa números con decimales.

```
precio = 19.99
```

Ilustración 3 Ejemplo de dato Decimal (float)

3. **Cadena de caracteres (str)**: Representa textos, que pueden incluir letras, números, espacios y otros caracteres.

```
nombre = "Ana María"
```

Ilustración 4 Ejemplo de Dato Cadena de Caracteres (str)

4. **Booleano (bool)**: Representa valores de verdad, es decir, True o False.

```
es_estudiante = True
es_vip = False
```

Ilustración 5 Ejemplo de Dato Booleano (bool)

Cada tipo de dato tiene sus propias operaciones y funciones asociadas. Por ejemplo, puedes realizar operaciones matemáticas con enteros y decimales, pero no puedes sumar una cadena de texto con un número directamente sin hacer conversiones.

EXPRESIONES ARITMÉTICAS EN PYTHON

Las **expresiones aritméticas** son operaciones matemáticas que podemos realizar utilizando variables y valores. Python soporta una variedad de operadores para realizar operaciones aritméticas:

- **Suma** (+)
- **Resta** (-)
- **Multiplicación** (*)
- **División** (/)
- **División entera** (//): Devuelve la parte entera de la división.
- **Módulo** (%): Devuelve el residuo de la división.
- **Exponenciación** (**): Eleva un número a una potencia.

```
a = 10
b = 3

suma = a + b  # Resultado: 13
resta = a - b  # Resultado: 7
multiplicacion = a * b  # Resultado: 30
division = a / b  # Resultado: 3.3333...
division_entera = a // b  # Resultado: 3
modulo = a % b  # Resultado: 1
exponente = a ** b  # Resultado: 1000
```

Ilustración 6 Ejemplo expresiones aritméticas con variables

CONVERSIONES DE TIPO

A veces es necesario convertir un tipo de dato a otro, lo que se conoce como **conversión de tipo** o **casting**. Python proporciona funciones incorporadas para convertir entre diferentes tipos de datos:

- **int()** convierte a entero.
- **float()** convierte a decimal.
- **str()** convierte a cadena de caracteres.
- **bool()** convierte a valor booleano.

```
# Conversión a entero
numero_decimal = 3.14
numero_entero = int(numero_decimal)  # Resultado: 3

# Conversión a cadena
numero_entero = 25
cadena = str(numero_entero)  # Resultado: "25"

# Conversión a booleano
valor = 0
es_true = bool(valor)  # Resultado: False
```

Ilustración 7 Ejemplo de casteo de datos

Es importante tener en cuenta que no todas las conversiones son posibles. Por ejemplo, intentar convertir una cadena que no representa un número a un entero generará un error.

IMPRESIÓN Y ENTRADA DE DATOS EN CONSOLA

En Python, puedes interactuar con el usuario a través de la consola utilizando las funciones **print()** e **input()**.

- **print()**: Muestra información en la consola.

```
nombre = "Carlos"
print("Hola, " + nombre)  # Imprime: Hola, Carlos
```

Ilustración 8 Ejemplo impresión en consola

- **input()**: Permite que el usuario ingrese datos. Todo lo que ingresa el usuario se convierte en una cadena de texto.

```
edad = input("¿Cuántos años tienes? ")
print("Tienes " + edad + " años.")
```

Ilustración 9 Ejemplo ingreso de datos

Recuerda que, si esperas que el usuario ingrese un número, debes convertirlo adecuadamente:

```
edad = int(input("¿Cuántos años tienes? "))  # Convierte la entrada a entero
```

Ilustración 10 Casteo de dato ingresado con input

CREACIÓN Y EJECUCIÓN DE UN SCRIPT PYTHON

Un **script Python** es simplemente un archivo de texto que contiene código Python y que tiene la extensión .py. Para crear y ejecutar un script, sigue estos pasos:

1. **Creación**:
    - Abre tu editor de texto o IDE (en este caso, Spyder, Jupyter, o cualquier editor que prefieras).

- Escribe tu código y guarda el archivo con la extensión .py. Por ejemplo, **mi_script.py**.

2. **Ejecución**:
   - Desde **Anaconda Navigator** o el terminal de tu sistema operativo, navega hasta la ubicación donde guardaste el archivo y ejecuta:

```
python mi_script.py
```

Ilustración 11 Ejemplo comando de ejecución

También puedes ejecutar el script desde el mismo editor que estés utilizando. Por ejemplo, en **Spyder** puedes hacer clic en el botón de "Ejecutar" o usar el atajo de teclado correspondiente.