Guía Práctica N.º 5 : Tipos de Datos y Operadores en Python

Resultado De Aprendizaje:

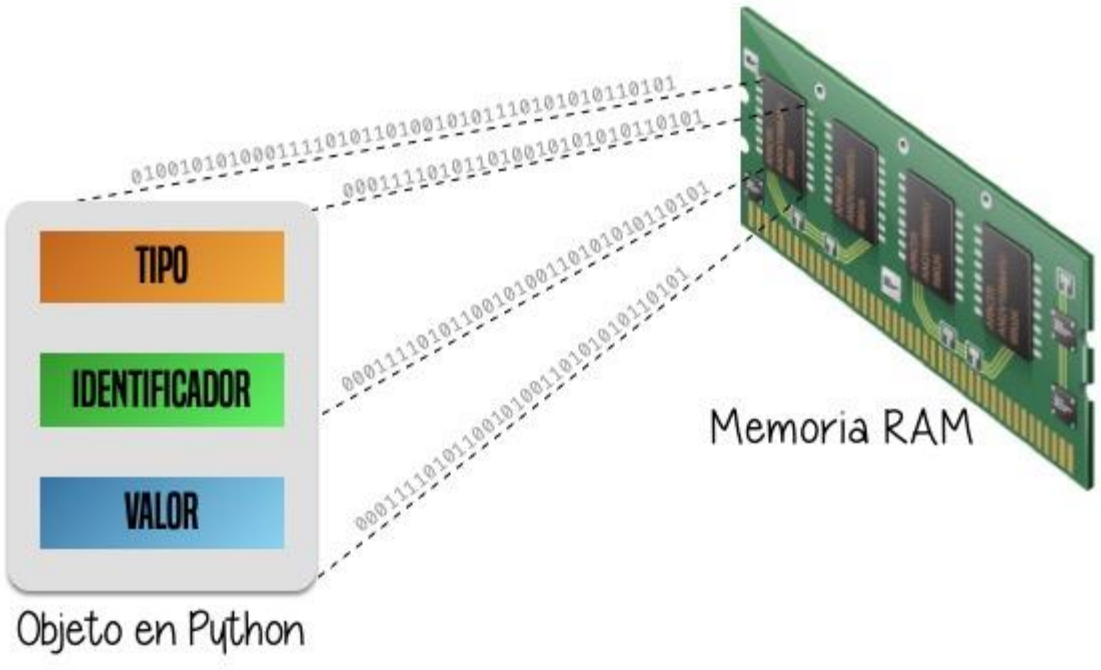
- Identificar los distintos tipos de datos a utilizar en la manipulación de datos en modo consola en Python
- Conocer los tipos de operadores en los distintos entornos de desarrollo para Python en modo consola
- Conocer los lineamientos básicos para usar los distintos tipos de operadores en programas de consola en Python.

Datos

Los programas están formados por **código** y **datos**. Pero a nivel interno de la memoria del ordenador no son más que una secuencia de bits. La interpretación de estos bits depende del lenguaje de programación, que almacena en la memoria no sólo el puro dato sino distintos metadatos.

Cada **«trozo»** de memoria contiene realmente un objeto, de ahí que se diga que en Python **todo son objetos**. Y cada objeto tiene, al menos, los siguientes campos:

- Un **tipo** del dato almacenado.
- Un **identificador** único para distinguirlo de otros objetos.
- Un **valor** consistente con su tipo.



Esquema (*metadatos*) de un objeto en Python

La lista más esencial de los más comunes Tipos de datos

A continuación, se muestran los distintos tipos de datos que podemos encontrar en Python, sin incluir aquellos que proveen paquetes externos:

Tipos de datos en Python		
Nombre	**Tipo**	**Ejemplos**
Booleano	bool	True, False
Entero	int	21, 34500
Flotante	float	3.14, 1.5e3
Complejo	complex	2j, 3 + 5j
Cadena	str	'tfn', '''tenerife - islas canarias'''
Tupla	tuple	(1, 3, 5)
Lista	list	['Chrome', 'Firefox']
Conjunto	set	set([2, 4, 6])
Diccionario	dict	{'Chrome': 'v79' , 'Firefox': 'v71'}

Es importante mencionar que las variables en Python no están sujetas a restricciones de tipo. En Python, una variable puede ser asignada con un valor de un tipo y luego reasignada con un valor de un tipo diferente.

Tipos de Datos en Python:

1. **Números enteros (int)**: Los números enteros son aquellos que no tienen parte decimal.
   Por ejemplo, 1, 2, 3, -4, etc. Se utilizan para realizar operaciones matemáticas como sumas, restas, multiplicaciones y divisiones.
2. **Números flotantes (float):** Los números flotantes son aquellos que tienen una parte decimal.
   Por ejemplo, 3.14, -0.5, etc. Al igual que los enteros, se utilizan para realizar operaciones matemáticas.
3. **Cadenas de texto (str):** Las cadenas de texto son secuencias de caracteres. Pueden contener letras, números y símbolos. Se utilizan para representar texto.
4. **Booleanos (bool):** Los booleanos son un tipo de dato que solo puede tener dos valores: verdadero (True) o falso (False). Se utilizan para representar la verdad o falsedad de una condición.
5. **Listas:** Las listas son una colección ordenada de elementos. Cada elemento puede ser de cualquier tipo de dato. Las listas son mutables, lo que significa que puedes agregar y eliminar elementos.
6. **Tuplas:** Las tuplas son similares a las listas, pero son inmutables. Esto significa que una vez que una tupla es creada, no puedes agregar ni eliminar elementos.
7. **Diccionarios:** Los diccionarios son una colección de pares clave-valor. Cada clave es única y se utiliza para acceder a su valor correspondiente.
8. **Conjuntos (set):** Los conjuntos son una colección de elementos únicos. Se utilizan cuando quieres almacenar múltiples elementos, pero te aseguras de que cada elemento solo aparezca una vez.

```python
numero_entero = 10
print(numero_entero)
numero_flotante = 3.14
print(numero_flotante)
cadena_texto = "¡Hola, mundo!"
print(cadena_texto)
valor_booleano = True
print(valor_booleano)
lista = [1, 2, 3, "cuatro", 5.0]
print(lista)
tupla = (1, 2, 3, "cuatro", 5.0)
print(tupla)
diccionario = {"nombre": "Juan", "edad": 30, "ciudad": "Madrid"}
print(diccionario)
conjunto = {1, 2, 3, 4, 5}
print(conjunto)
```

OPERADORES

Los operadores en Python son símbolos especiales que realizan operaciones en variables y valores. Python divide los operadores en los siguientes grupos:

1. **Operadores aritméticos**: Se utilizan con valores numéricos para realizar operaciones matemáticas comunes como suma (+), resta (-), multiplicación (*), división (/), módulo (%), exponenciación (**) y división de piso (//).

2. **Operadores de asignación**: Se utilizan para asignar valores a las variables. Estos incluyen el operador de asignación (=) y los operadores de asignación compuesta como +=, -=, *=, /=, %=, //=, **=, &=, |=, ^=, >>=, <<=.

3. **Operadores de comparación**: Se utilizan para comparar dos valores.

   Incluyen igual (==), no igual (!=), mayor que (>), menor que (<), mayor o igual que (>=), y menor o igual que (<=).

4. **Operadores lógicos**: Se utilizan para combinar declaraciones condicionales. Incluyen "and", "or" y "not".

5. **Operadores de identidad**: Se utilizan para comparar los objetos, no si son iguales, sino si son realmente el mismo objeto, con la misma ubicación de memoria. Incluyen "is" e "is not".

6. **Operadores de membresía**: Se utilizan para probar si una secuencia está presente en un objeto. Incluyen "in" y "not in".

7. **Operadores a nivel de bits**: Se utilizan para comparar (números binarios). Incluyen AND (&), OR (|), XOR (^), NOT (~), desplazamiento a la izquierda (<<) y desplazamiento a la derecha (>>).

Algunos de los ***operadores más utilizados*** en Python son los siguientes los cuales resalta su uso en modo consola:

1. **Operadores aritméticos**: Estos operadores se utilizan para realizar operaciones aritméticas como suma (+), resta (-), multiplicación (*), división (/), división entera (//), módulo (%) y exponenciación (**).
2. **Operadores de comparación**: Estos operadores se utilizan para comparar valores. Incluyen igual a (==), diferente de (!=), menor que (<), mayor que (>), menor o igual que (<=), y mayor o igual que (>=).
3. **Operadores lógicos**: Estos operadores se utilizan para combinar declaraciones condicionales. Incluyen Y lógico (and), O lógico (or), y No lógico (not).
4. **Operadores de asignación**: Estos operadores se utilizan para asignar valores a las variables. Incluyen asignación (=), suma en asignación (+=), resta en asignación (-=), multiplicación en asignación (*=), división en asignación (/=), división entera en asignación (//=), módulo en asignación (%=), y exponenciación en asignación (**=).

Ejemplos:

1. **Operadores aritméticos**:

```python
x = 10
y = 3

print(x + y)  # Suma: 13
print(x - y)  # Resta: 7
print(x * y)  # Multiplicación: 30
print(x / y)  # División: 3.3333333333333335
print(x // y) # División entera: 3
print(x % y)  # Módulo: 1
print(x ** y) # Exponenciación: 1000
```

2. **Operadores de comparación:**

```python
x = 10
y = 3

print(x == y)  # Igual a: False
print(x != y)  # Diferente de: True
print(x < y)   # Menor que: False
print(x > y)   # Mayor que: True
print(x <= y)  # Menor o igual que: False
print(x >= y)  # Mayor o igual que: True

```

3. **Operadores lógicos:**

```python
x = True
y = False

print(x and y)  # Y lógico: False
print(x or y)   # O lógico: True
print(not x)    # No lógico: False
```

4. **Operadores de asignación:**

```python
x = 10  # Asignación: x es 10
x += 3  # Suma en asignación: x es ahora 13
x -= 3  # Resta en asignación: x es ahora 10
x *= 3  # Multiplicación en asignación: x es ahora 30
x /= 3  # División en asignación: x es ahora 10.0
x //= 3 # División entera en asignación: x es ahora 3.0
x %= 3  # Módulo en asignación: x es ahora 0.0
x **= 3 # Exponenciación en asignación: x es ahora 0.0
```

GUIA DE EJERCICIOS.

1. **Uso de enteros y operadores aritméticos**: Escribe un programa que pida al usuario dos números enteros e imprima su suma, resta, multiplicación y división. El programa debe solicitar al usuario que introduzca dos números enteros y luego debe calcular e imprimir la suma, resta, multiplicación y división de estos números.

2. **Uso de cadenas de texto y operadores de concatenación**: Escribe un programa que pida al usuario su nombre y su ciudad de origen e imprima un mensaje de bienvenida. El programa debe solicitar al usuario que introduzca su nombre y su ciudad de origen. Luego, debe generar un mensaje de bienvenida que incluya el nombre y la ciudad de origen del usuario.

3. **Uso de booleanos y operadores lógicos**: Escribe un programa que pida al usuario dos valores booleanos e imprima su conjunción, disyunción y negación. El programa debe solicitar al usuario que introduzca dos valores booleanos. Luego, debe calcular e imprimir la conjunción (y lógico), disyunción (o lógico) y negación (no lógico) de estos valores.

4. **Uso de listas y operadores de lista**: Escribe un programa que cree una lista con los nombres de tres amigos y añada un cuarto nombre al final. El programa debe crear una lista con los nombres de tres amigos. Luego, debe agregar un cuarto nombre al final de la lista e imprimir la lista.

5. **Uso de diccionarios y operadores de diccionario**: Escribe un programa que cree un diccionario con la edad y la ciudad de origen de una persona e imprima su edad. El programa debe crear un diccionario que contenga la edad y la ciudad de origen de una persona. Luego, debe imprimir la edad de la persona.