Tema 3: Operaciones Básicas con Variables

Lectura fácil

Operaciones aritméticas

Las **operaciones aritméticas** son las más comunes cuando trabajamos con variables numéricas en Python. A continuación, veremos las operaciones más básicas que puedes realizar.

Suma (+)

Puedes sumar dos números con el símbolo **+**.

Ejemplo:

```
a = 5
b = 3
resultado = a + b
print(resultado)
```

- En este ejemplo, sumamos 5 y 3, guardando el resultado en la variable **resultado**.
- El programa mostrará:
  8

Resta (-)

Puedes restar números utilizando el símbolo **-**.

Ejemplo:

```
a = 10
b = 4
resultado = a - b
print(resultado)
```

- Aquí estamos restando 4 de 10.

- El programa mostrará:
  6

Multiplicación (*)

Para multiplicar dos números, se usa el símbolo ***.

Ejemplo:

```
a = 6
b = 7
resultado = a * b
print(resultado)
```

- Multiplicamos 6 por 7.
- El programa mostrará:
  42

División (/)

Para dividir, usamos el símbolo */*.

Ejemplo:

```
a = 8
b = 2
resultado = a / b
print(resultado)
```

- Dividimos 8 entre 2.
- El programa mostrará:
  4.0

Nota: Cuando usas la división, Python siempre devuelve un número con decimales (**float**), incluso si el resultado es un número entero. Por ejemplo, 8 dividido entre 2 es 4, pero Python lo muestra como **4.0**.

División entera (//)

Si quieres que Python te devuelva solo la parte entera del resultado (sin decimales), puedes usar *//*.

Ejemplo:

```
a = 9
b = 4
resultado = a // b
print(resultado)
```

- Aquí, 9 dividido entre 4 da **2.25**, pero Python te devuelve solo el número entero **2**.
- El programa mostrará:
  2

Módulo (%)

El operador **%** devuelve el **resto** de una división. Es muy útil cuando quieres saber si un número es divisible por otro o si hay un "sobrante".

Ejemplo:

```
a = 10
b = 3
resultado = a % b
print(resultado)
```

- Cuando divides 10 entre 3, el resultado es 3 con un **resto** de 1.
- El programa mostrará:
  1

Potencia ()**

Para elevar un número a una potencia, usamos **.

Ejemplo:

```
a = 2
b = 3
resultado = a ** b
```

```
print(resultado)
```

- Estamos elevando 2 a la potencia de 3 (2 * 2 * 2).
- El programa mostrará:
  8

Operaciones con cadenas de texto

Además de números, también puedes realizar operaciones con **cadenas de texto**. Las cadenas de texto en Python son secuencias de caracteres entre comillas, como `"Hola"` o `"Python"`.

Concatenación (+)

En Python, puedes unir (concatenar) dos o más cadenas de texto utilizando el símbolo **+**.

Ejemplo:

```
nombre = "Juan"
apellido = "Pérez"
nombre_completo = nombre + " " + apellido
print(nombre_completo)
```

- Aquí estamos uniendo el nombre "Juan" y el apellido "Pérez" con un espacio en medio.
- El programa mostrará:
  Juan Pérez

Repetición (*)

Puedes repetir una cadena de texto varias veces usando el símbolo ***.

Ejemplo:

```
frase = "Hola! "
repetida = frase * 3
print(repetida)
```

- En este caso, repetimos la frase "Hola! " tres veces.
- El programa mostrará:
  Hola! Hola! Hola!

Acceder a un carácter de una cadena

Cada carácter en una cadena tiene un **índice** (una posición) y puedes acceder a un carácter específico usando corchetes [ ]. En Python, los índices empiezan desde **0**.

Ejemplo:

```
frase = "Python"
letra = frase[0]
print(letra)
```

- Estamos accediendo al primer carácter de la cadena "Python", que es **P**.
- El programa mostrará:
  P

Longitud de una cadena (len)

Puedes saber cuántos caracteres tiene una cadena utilizando la función **len()**.

Ejemplo:

```
frase = "Python"
longitud = len(frase)
print(longitud)
```

- Estamos obteniendo el número de caracteres en la palabra "Python".
- El programa mostrará:
  6

Operaciones con listas

Una **lista** es una colección de elementos que pueden ser de diferentes tipos, como números, cadenas de texto, o incluso otras listas. Las listas te permiten agrupar varios valores y manipularlos fácilmente.

Crear una lista

Para crear una lista en Python, simplemente usa corchetes [ ] y separa los elementos con comas.

Ejemplo:

```
mi_lista = [1, 2, 3, 4, 5]
print(mi_lista)
```

- Este código crea una lista de números.
- El programa mostrará:
  [1, 2, 3, 4, 5]

Acceder a elementos de una lista

Puedes acceder a un elemento de una lista utilizando su **índice**, como hicimos con las cadenas de texto.

Ejemplo:

```
mi_lista = ["manzana", "banana", "cereza"]
print(mi_lista[1])
```

- Aquí estamos accediendo al segundo elemento de la lista (recuerda que los índices empiezan desde 0).
- El programa mostrará:
  banana

Añadir elementos a una lista (append)

Puedes agregar un nuevo elemento a una lista usando el método **append()**.

Ejemplo:

```
mi_lista = ["manzana", "banana"]
mi_lista.append("cereza")
```

```
print(mi_lista)
```

- El programa agregará "cereza" al final de la lista.
- El programa mostrará:
  ["manzana", "banana", "cereza"]

Eliminar elementos de una lista (remove)

Si quieres eliminar un elemento de una lista, puedes usar el método **remove()**.

Ejemplo:

```
mi_lista = ["manzana", "banana", "cereza"]
mi_lista.remove("banana")
print(mi_lista)
```

- Este código eliminará "banana" de la lista.
- El programa mostrará:
  ["manzana", "cereza"]

Las operaciones básicas con variables son fundamentales para escribir programas en Python. Hemos aprendido cómo realizar **operaciones aritméticas** con números, cómo **manipular cadenas de texto**, y cómo **trabajar con listas**. Todas estas herramientas son esenciales para resolver problemas y escribir código más complejo.
