🐍 Clase 04 - Métodos de String y operadores aritméticos.

Strings y métodos asociados

¿Qué obtendremos al "sumar" dos Strings?

```
a = 'HOLA '
b = 'MUNDO'
print(a + b)
```

```
HOLA MUNDO
```

> **Al asignar los valores utilizando comillas simples, Python los interpreta como String. Por ende, la instrucción dada por el carácter `+`, para este caso, es de concatenar, no de sumar.**

En programación, la acción de 'sumar' dos o más Strings se conoce como *concatenación*.

Concatenación

Podemos obtener los mismo resultados utilizando concatenación e interpolación, sin embargo, se prefiere la interpolación debido a que es más rápida y presenta una sintaxis más amigable para el desarrollador.

```
nombre = 'Carlos'
apellido = 'Santana'
# Concatenación
print("Mi nombre es " + nombre + " " + apellido)
```

```
Mi nombre es Carlos Santana
```

Interpolación

Otra acción muy importante y ampliamente utilizada al momento de trabajar con Strings es la interpolación.

La interpolación es un mecanismo que nos permite introducir una variable (o un dato) dentro un String, sin necesidad de concatenarlo. Para interpolar simplemente tenemos que introducir la variable (o dato) utilizando la siguiente notación:

```
nombre = 'Carlos'
apellido = 'Santana'
# Interpolación
print("Mi nombre es {} {}".format(nombre, apellido))
```

```
Mi nombre es Carlos Santana
```

📌 f-strings en Python

Las **f-strings** (formatted string literals) se introdujeron en Python 3.6 y permiten insertar variables o expresiones directamente dentro de una cadena de texto.

🔹 Sintaxis básica

```
f"texto {variable}"
```

O también con comillas simples:

```
f'texto {variable}'
```

👉 La clave es anteponer la letra `f` (o `F`) antes de las comillas.

🔹 Ejemplo simple

```
nombre ="Juan"
edad =35

mensaje =f"Me llamo {nombre} y tengo{edad} años."
print(mensaje)
```

Salida:

```
Me llamo Juany tengo 35años.
```

🔹 Insertar expresiones (no solo variables)

Puedes meter operaciones directamente dentro de `{}`:

```
precio = 100
iva = 0.21

total = f"El total con IVA es {precio * (1 + iva)} euros"
print(total)
```

🔹 Formateo de números

La sintaxis general para formatear es:

```
f"{variable:formato}"
```

✅ Decimales

```
pi =3.14159265
print(f"{pi:.2f}")
```

Salida:

```
3.14
```

✅ Separador de miles

```
numero =1000000
print(f"{numero:,}")
```

Salida:

```
1,000,000
```

🔹 Alineación

```
texto ="Python"

print(f"{texto:<10}")# izquierda
print(f"{texto:>10}")# derecha
print(f"{texto:^10}")# centrado
```

🔹 Mostrar el nombre de la variable (debug, Python 3.8+)

```
x = 10
print(f"{x=}")
```

Salida:

```
x=10
```

Muy útil para depuración.

📌 Resumen general de la sintaxis

```
f"texto {variable}"
f"texto {expresión}"
f"{variable:formato}"
f"{variable:ancho.formato}"
```

Count

Otro método que se puede aplicar a las variables de tipo string es `count`, que nos permite contar la cantidad de ocurrencias de un carácter específico dentro de un texto.

En este caso, al hacer:

```
nombre = 'Carlos'
apellido = 'Santana'
print(nombre.count("a"))
```

```
Obtenemos como resultado "1", porque la letra "a" aparece 1 v
ez en el texto "Carlos".
```

Len()

Entrega la cantidad de letras o el "largo" del texto. Por tanto al ejecutar:

```
nombre = 'Carlos'
apellido = 'Santana'
print(len(apellido))
```

```
Se obtiene 7, que es la cantidad de letras de "Santana".
```

Upper()

Aplica letras mayúsculas a todo el texto:

```
nombre = 'Carlos'
apellido = 'Santana'
print(apellido.upper())
```

```
SANTANA
```

El salto de línea en un string

El salto de línea `\n` es un caracter especial que nos permite agregar un salto de línea dentro de un string.

```
saltos = "hola\na\ntodos"
print(saltos)
```

```
hola
a
todos
```

Tabulación horizontal \t (horizontal tab)

```
print("Hello\tWorld")
```

```
Hello    World
```

La función `input()` y entrada de datos por teclado en Python

La función `input()` en Python permite que un programa reciba datos ingresados por el usuario a través del teclado. Es una herramienta fundamental para hacer que los programas sean interactivos.

- **Sintaxis básica**:

  ```
  variable = input("Mensaje para el usuario: ")
  ```

- El mensaje dentro de `input()` se muestra en pantalla para guiar al usuario.
- Todo lo que el usuario ingresa se almacena como una **cadena de texto** (string) en la variable.

Ejemplo 1:

```
nombre = input("¿Cuál es tu nombre? ")
print("¡Hola, " + nombre + "!")
```

Salida (si el usuario ingresa "Ana"):

```
¿Cuál es tu nombre? Ana
¡Hola, Ana!
```

> **Punto clave**: La función `input()` siempre devuelve un string, incluso si el usuario ingresa números. Si necesitas un número, debes convertirlo.

---

2. Operadores Aritméticos y Relacionales

Sintaxis

Operadores Aritméticos

```
# Operadores básicos
+   # Suma
-   # Resta
*   # Multiplicación
/   # División
//  # División entera 7 // 2
= 3
%   # Módulo (resto)
**  # Potencia
```

Operadores Relacionales

```
==  # Igual a
!=  # Diferente de
>   # Mayor que
<   # Menor que
>=  # Mayor o igual que
<=  # Menor o igual que
```

Ejemplos Básicos Resueltos

Ejemplo 1: Operaciones aritméticas básicas

```
# Operaciones con dos números
a = 15
b = 4

suma = a + b           # 19
resta = a - b          # 11
multiplicacion = a * b # 60
division = a / b       # 3.75
division_entera = a // b # 3
modulo = a % b        # 3
potencia = b ** 2     # 16

print(f"15 + 4 = {suma}")
print(f"15 - 4 = {resta}")
print(f"15 * 4 = {multiplicac
ion}")
```

Ejemplo 2: Comparaciones numéricas

```
# Comparar dos valores
x = 25
y = 30

print(f"x = {x}, y = {y}")
print(f"x == y: {x == y}")  #
False
print(f"x != y: {x != y}")  #
True
print(f"x > y: {x > y}")    #
False
print(f"x < y: {x < y}")    #
True
print(f"x >= 25: {x >= 25}")
# True
```

```
print(f"15 / 4 = {division}")
print(f"15 // 4 = {division_entera}")
print(f"15 % 4 = {modulo}")
print(f"4 ** 2 = {potencia}")
```

```
print(f"y <= 30: {y <= 30}")
# True
```

Ejemplo 3: Cálculo de promedio

```
# Calcular el promedio de tres números
nota1 = 85
nota2 = 92
nota3 = 78

suma_total = nota1 + nota2 + nota3
promedio = suma_total / 3
# promedio = (nota1 + nota2 + nota3)/3
print(f"Notas: {nota1}, {nota2}, {nota3}")
print(f"Suma total: {suma_total}")
print(f"Promedio: {promedio:.2f}")

# Verificar si aprobó (promedio >= 70)
aprobado = promedio >= 70
print(f"¿Aprobó? {aprobado}")
# Modificar incluyendo input()
```

Ejemplo 4: Verificación de número par

```
# Determinar si un número es par o impar
numero = 42

residuo = numero % 2
es_par = residuo == 0

print(f"Número: {numero}")
print(f"Residuo al dividir entre 2: {residuo}")
print(f"¿Es par? {es_par}")

# Verificación adicional
es_multiplo_de_3 = numero % 3 == 0
print(f"¿Es múltiplo de 3? {es_multiplo_de_3}")
# Modificar incluyendo input()
```

Ejercicios para Practicar

Ejercicio 1: Calculador de edad en días

Enunciado: Crea un programa que calcule la edad aproximada en días y horas. Dado que una persona tiene 23 años, calcula cuántos días y horas ha vivido aproximadamente (considera 365 días por año). Verifica si es mayor de edad (>= 18 años).

Salida esperada:

```
Edad: 23 años
Días vividos: 8395 días
Horas vividas: 201480 horas
¿Es mayor de edad? True
```

Ejercicio 2: Conversor de temperatura

Enunciado: Desarrolla un programa que convierta 28 grados Celsius a Fahrenheit usando la fórmula F = C * 9/5 + 32. Verifica si la temperatura en Fahrenheit es mayor a 80°F (clima cálido).

Salida esperada:

```
Temperatura en Celsius: 28°C
Temperatura en Fahrenheit: 82.40°F
¿Es clima cálido (>80°F)? True
Diferencia con punto de congelación: 50.40°F
```

Ejercicio 3: Análisis de ventas

Enunciado: Calcula las estadísticas de ventas de una tienda. Las ventas de 3 días fueron: $1250, $980, $1430. Calcula el total, el promedio diario y verifica si el promedio supera la meta de $1200.

Salida esperada:

```
Ventas día 1: $1250
Ventas día 2: $980
```

```
Ventas día 3: $1430
Total de ventas: $3660
Promedio diario: $1220.00
¿Supera la meta diaria ($1200)? True
```