## 2. Variables y tipos de datos

En la programación, las variables son esenciales ya que permiten almacenar y manipular datos en la memoria. Python es un lenguaje de programación con una tipificación dinámica, lo que significa que no es necesario declarar el tipo de una variable al momento de su creación. Además, Python ofrece una gran variedad de tipos de datos incorporados, tales como números, cadenas, booleanos, listas, tuplas, diccionarios y más.

En este tema, aprenderemos cómo crear, declarar, inicializar y trabajar con variables en Python. También exploraremos los diferentes tipos de datos disponibles, y cómo convertir y manipularlos para realizar operaciones útiles

### a) Variables

Las variables son una parte importante de cualquier lenguaje de programación. En Python, una variable es un espacio en la memoria que se utiliza para almacenar valores que se pueden cambiar a lo largo del programa.

En Python, una variable es un nombre que se utiliza para hacer referencia a un valor en particular. Para declarar una variable, utilizamos el signo igual “=” y le asignamos un valor. Por ejemplo, si queremos declarar una variable llamada “edad” y le asignamos un valor de 25, escribimos

```
edad = 25
```

Es importante tener en cuenta que una variable puede cambiar de valor durante la ejecución del programa.

La inicialización es el proceso de asignar un valor a una variable por primera vez. Por ejemplo, si en un programa tenemos una variable llamada “nombre” y en un determinado momento le asignamos el valor “Juan”, decimos que hemos inicializado la variable “nombre” con el valor “Juan”.

### b) Tipos de datos

En Python, existen varios tipos de datos que podemos utilizar. Los más comunes son:

- **Enteros**: números enteros sin decimales

```
x = 5
y = 12345
```

```
z = -10
```

- **Flotantes**: números con decimales

```
a = 3.14
b = -2.5
c = 1.0
```

- **Cadenas de caracteres**: secuencia de caracteres que representan texto,

```
nombre = "Juan"
apellido = 'Pérez'
saludo = "¡Hola, ¿cómo estás?"
```

- **Booleanos**: valores verdadero o falso, como True o False.

```
verdadero = True
falso = False
resultado = 5 > 10
print(resultado)
```

```
> False
```

### c) Conversión y casting de tipos

En ocasiones, es necesario convertir un tipo de dato a otro para realizar operaciones o comparaciones. Para ello, Python nos ofrece las funciones int(), float(), str() y bool().

Por ejemplo, si tenemos un número representado como una cadena de caracteres, podemos convertirlo en un número entero utilizando la función int(). Por ejemplo:

```
x = "5"
y = int(x)
```

En este caso, la variable "y" contendría el valor entero 5.

### d) Operaciones de cadenas de caracteres

En Python, podemos realizar diversas operaciones con cadenas de caracteres, tales como:

- **Concatenación:** unir dos cadenas de caracteres utilizando el operador **+**.

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

```
> Juan Pérez
```

- **Repetición:** repetir una cadena varias veces utilizando el operador *****.

```
cadena = "Hola "
repetido = cadena * 3
print(repetido)
```

```
> Hola Hola Hola
```

- **Segmentación:** obtener una porción de una cadena de caracteres utilizando los índices **[inicio : fin]**.

```
cadena = "Hola mundo"
subcadena = cadena[0:4]
print(subcadena)
```

```
> Hola
```

- **Indexación:** obtener un carácter específico de una cadena de caracteres utilizando su índice.

```
cadena = "Hola mundo"
primer_caracter = cadena[0]
print(primer_caracter)
```

```
> H
```

### e) Métodos de cadenas de caracteres

Además de las operaciones básicas, Python también ofrece una gran variedad de métodos para manipular cadenas de caracteres. Algunos de los más comunes son:

- **upper():** convierte todos los caracteres de una cadena de caracteres a mayúsculas.

```
cadena = "hola"
mayusculas = cadena.upper()
print(mayusculas)
```

```
> HOLA
```

- **lower():** convierte todos los caracteres de una cadena de caracteres a minúsculas.

```
cadena = "MANZANA ROJA"
minusculas = cadena.lower()
print(minusculas)
```

```
> manzana roja
```

- **strip():** elimina los espacios en blanco al inicio y al final de una cadena de caracteres.

```
cadena = "   hola mundo   "
sin_espacios = cadena.strip()
print(sin_espacios)
```

```
> hola mundo
```

- **replace():** reemplaza una subcadena de una cadena de caracteres por otra subcadena.

```
cadena = "Hola Mundo"
reemplazo = cadena.repace("Mundo","Python")
print(sin_espacios)
```

```
> Hola Python
```

- **split():** separa una cadena de caracteres en una lista de subcadenas utilizando un separador especificado.

```
cadena = "Hola Mundo"
lista = cadena.split(" ")
print(lista)
```

```
> ["Hola","Mundo"]
```

En resumen, en este tema hemos visto cómo declarar y utilizar variables en Python, los distintos tipos de datos que podemos manejar, cómo convertir entre ellos y cómo realizar operaciones y manipulaciones de cadenas de caracteres. Estos conceptos son fundamentales en la programación y nos permiten construir programas más complejos y sofisticados.