



ProfesorenVideo.com

# Tipos de datos en Python

Text Type: `str`

Numeric Types: `int`, `float`, `complex`

Sequence Types: `list`, `tuple`, `range`

Mapping Type: `dict`

Set Types: `set`, `frozenset`

Boolean Type: `bool`

Binary Types: `bytes`, `bytearray`, `memoryview`

None Type: `NoneType`



ProfesorenVideo.com

# Tipos de datos en Python

TIPO DE DATO str (STRING) O CADENA: Permite representar combinación de letras, números y caracteres especiales. Para su manejo debemos encerrarlos entre comillas dobles o comilla simple.

## Ejemplos:

"Python es multi propósito"

"Python tiene una curva de aprendizaje bastante plana"

'Calle los Olivos, casa # 22, al lado de la estación 1'

'+56-987231541'









ProfesorenVideo.com

# Tipos de datos en Python

TIPO DE DATO **float** (FLOTANTE) O DECIMAL: Permite representar números con parte decimal. A diferencia de los enteros si tienen límites máximos y mínimos.

La mínima precisión es **2.2250738585072014e-308** y la máxima **1.7976931348623157e+308**.

Ejemplos:

42.7

12.34

-156.28

43.286





# Tipos de datos en Python

LISTAS , TUPLAS, CONJUNTOS y DICCIONARIOS :

**LISTAS** (**list**): Las listas se utilizan para almacenar varios elementos en una sola variable. La lista es una colección ordenada y modificable. Permite miembros duplicados.

**TUPLAS** (**tuple**): Tupla es una colección ordenada e inmutable. Permite miembros duplicados.

\* Los elementos establecidos no se pueden cambiar, pero puedes eliminar y/o agregar elementos cuando quieras.



# Tipos de datos en Python

## LISTAS , TUPLAS, CONJUNTOS y DICCIONARIOS :

**LISTAS (list):** Las listas se utilizan para almacenar varios elementos en una sola variable. La lista es una colección *ordenada* y *modificable*. Permite miembros duplicados. Se escriben con `[]`.

```
lista_colores = ["rojo", "blanco", "verde"]  
print(lista_colores)
```

Las listas permiten duplicados:

```
lista_colores = ["rojo", "blanco", "verde", "rojo"]  
print(lista_colores)
```

Los elementos de la lista están indexados. Requieren un índice para acceder a ellos.

`Lista_colores[0]` # primer elemento.

`Lista_colores[1]` # segundo elemento.

Cuando decimos que las listas están ordenadas, significa que los elementos tienen un orden definido y ese orden se mantiene, no cambia.

Si se agrega nuevos elementos a una lista, los nuevos elementos se colocarán al final de la lista.





ProfesorenVideo.com

# Tipos de datos en Python

**LISTA (list):** Son indexadas, permiten duplicados, se les puede agregar y eliminar elementos, tienen longitud.

Las listas permiten duplicados:

```
lista_colores = ["rojo", "blanco", "verde", "rojo"]  
print(lista_colores)
```

La longitud de una lista se puede obtener con el método: `len()`

```
lista_colores = ["rojo", "blanco", "verde", "rojo"]  
print(len(lista_colores)) # 4
```

Cuando decimos que las listas están ordenadas, significa que los elementos tienen un orden definido y ese orden se mantiene, no cambia.

Si agrega nuevos elementos a una lista, los nuevos elementos se colocarán al final de la lista.





ProfesorenVideo.com

# Tipos de datos en Python

**TUPLAS (tuple):** Se utilizan para almacenar varios elementos en una sola variable. Una tupla es una colección ordenada e inmutable. Las tuplas se escriben entre paréntesis `()`. Son ordenadas y permiten duplicados.

```
tupla_colores = ("rojo", "blanco", "verde")  
print(tupla_colores)
```

Los elementos de tupla están indexados, el primer elemento tiene el índice `[0]`, el segundo elemento tiene el índice `[1]`, etc.

Las tuplas no se pueden cambiar, lo que significa que no podemos cambiar, agregar o eliminar elementos una vez creada la tupla.

Los elementos de la lista están indexados. Requieren un índice para acceder a ellos.

`Lista_colores[0]` # primer elemento.

`Lista_colores[1]` # segundo elemento.

Cuando decimos que las tuplas están ordenadas, significa que los elementos tienen un orden definido y ese orden se mantiene, no cambia.





ProfesorenVideo.com

# Tipos de datos en Python

**CONJUNTOS (set):** Los conjuntos se utilizan para almacenar varios elementos en una sola variable. Un conjunto es una colección *desordenada*, *inmutable\** y *no indexada*. Se escriben entre `{ }`. No permite duplicados.

\* Nota: Los elementos tipo conjunto no se pueden cambiar, pero se pueden eliminar elementos y agregar elementos nuevos.

```
conjunto_paises= {"Venezuela", "Chile", "Bolivia"}  
print(conjunto_paises)
```

*Desordenado* significa que los elementos de un conjunto no tienen un orden definido. Los elementos de conjunto pueden aparecer en un *orden diferente* cada vez que los usa y no se puede hacer referencia a ellos mediante índice o clave.

Una vez que se crea un conjunto, *no se pueden cambiar sus elementos*, pero **SÍ** se puede eliminar elementos y agregar elementos nuevos, siempre al final.

```
conjunto= {"avión", "barco", True, False, 1, 0, 6}
```

**Nota:** Los valores **True** y **1** se consideran el mismo valor en conjuntos y se tratan como duplicados. Igual sucede con **False** y **0**.





ProfesorenVideo.com

# Tipos de datos en Python

**DICCIONARIO (dict):** Los diccionarios se utilizan para almacenar valores de datos en pares **clave : valor**. Un diccionario es una colección ordenada, modificable y que no permite duplicados. En la versión **3.7** de Python, los diccionarios están ordenados. En Python 3.6 y versiones anteriores están desordenados. Los diccionarios están escritos entre **{ }**.

```
diccionario_autos = {  
    "marca" : "Ford",  
    "modelo" : "Mustang",  
    "anio" : 1964  
}
```

Los elementos del diccionario están ordenados, son modificables y no permiten duplicados. Los elementos del diccionario se presentan en pares **clave:valor** y se puede hacer referencia a ellos mediante el nombre de la clave.

El diccionario puede tener datos de diferentes tipos:

```
diccionario = {  
    "marca": "Ford",  
    "electrico": False,  
    "anio": 1964,  
    "colores": ["blanco", "azul", "negro"]  
}  
print ( diccionario )  
  
print ( diccionario[marca] )
```





ProfesorenVideo.com

# Tipos de datos en Python

**BOOLEANOS** (**bool**): El tipo de dato booleano representa uno de dos valores: **True** (verdad) o **False** (falso). En Python cualquier valor entero distinto de cero (0) se asume como verdadero, y el cero (0) como falso.

## Ejemplos:

```
a = True  
b = False  
c = 9 < 1  
d = 8 > 4  
e = 15 == 15
```





ProfesorenVideo.com

*Muchas gracias por su atención*