

Tipos Básicos

Enteros

```
In [2]: 2 + 4 # suma
```

```
Out[2]: 6
```

```
In [3]: 3 * 5 # multiplicación
```

```
Out[3]: 15
```

```
In [4]: 4 - 7 # resta de números enteros
```

```
Out[4]: -3
```

```
In [5]: 8 // 3 # división entera, cociente
```

```
Out[5]: 2
```

```
In [6]: 8 % 3 # división entera, resto
```

```
Out[6]: 2
```

```
In [7]: 2 + 4 * 3 # la multiplicación tiene prioridad
```

```
Out[7]: 14
```

```
In [8]: (2 + 4) * 3 # los paréntesis cambian esa prioridad
```

```
Out[8]: 18
```

```
In [9]: 2 ** 3 # potencia
```

```
Out[9]: 8
```

```
In [10]: a = 5 # utilizamos el nombre a para acceder a un valor  
b = a * 3 # podemos utilizar los nombres en las expresiones  
a = 2 * a # se evalúa la expresión y se modifica el valor almacenado en a  
a, b
```

```
Out[10]: (10, 15)
```

```
In [11]: n = 7  
(n * (n + 1)) // 2 # los paréntesis no hacen falta en esta expresión
```

```
Out[11]: 28
```

```
In [12]: def poli(x):  
         c2 = 3 * x**2  
         c1 = -4 * x  
         c0 = -6  
         return c2 + c1 + c0  
  
poli(5)
```

Out[12]: 49

```
In [13]: def poli(x):  
         s = 0  
         pot = 1  
         s = s + (-6 * pot)  
         pot = pot * x  
         s = s + (-4 * pot)  
         pot = pot * x  
         s = s + (3 * pot)  
         return s  
  
poli(5)
```

Out[13]: 49

Reales, aritmética de coma flotante

```
In [14]: a = 2.0 # El punto indica la coma decimal  
         b = 3.0  
         a + b # suma
```

Out[14]: 5.0

```
In [15]: a * 7.0 # multiplicación
```

Out[15]: 14.0

```
In [16]: a / 6.0 # división entre reales
```

Out[16]: 0.3333333333333333

```
In [17]: a * 7 # los enteros se convierten en reales automáticamente
```

Out[17]: 14.0

```
In [18]: def poli(x):  
         s = 0.0  
         pot = 1  
         s = s + (-6 * pot)  
         pot = pot * x  
         s = s + (-4 * pot)  
         pot = pot * x  
         s = s + (3 * pot)  
         return s
```

```
poli(5)
```

```
Out[18]: 49.0
```

```
In [19]: 2.0 ** 3.1 # potencia
```

```
Out[19]: 8.574187700290345
```

```
In [20]: import math # podemos usar muchas funciones matemáticas
```

```
In [21]: radio = 3  
         2 * radio * math.pi # math.pi es un nombre que contiene el número pi
```

```
Out[21]: 18.84955592153876
```

```
In [22]: math.sin(math.pi/3) # math.sin, el seno del ángulo en radianes
```

```
Out[22]: 0.8660254037844386
```

```
In [23]: math.cos(math.pi/3) # math.cos, el coseno. Obsérvese el error obtenido
```

```
Out[23]: 0.5000000000000001
```

```
In [24]: math.sqrt(5) # la raíz cuadrada
```

```
Out[24]: 2.23606797749979
```

```
In [25]: int(3.9) # construye un entero truncando la parte decimal
```

```
Out[25]: 3
```

```
In [26]: int(-3.9)
```

```
Out[26]: -3
```

```
In [27]: round(3.2) # redondea al entero más próximo. El resultado sigue siendo real
```

```
Out[27]: 3.0
```

```
In [28]: round(3.5)
```

```
Out[28]: 4.0
```

```
In [29]: round(-3.5)
```

```
Out[29]: -4.0
```

```
In [30]: round(-3.2)
```

```
Out[30]: -3.0
```

```
In [31]: math.floor(3.7) # redondea al entero inferior. El resultado sigue siendo real
```

```
Out[31]: 3.0
```

```
In [32]: math.floor(-3.2)
```

```
Out[32]: -4.0
```

```
In [33]: math.ceil(3.2)
```

```
Out[33]: 4.0
```

```
In [59]: math.ceil(-3.2)
```

```
Out[59]: -3.0
```

Booleanos

```
In [34]: a = True # valor lógico de cierto  
b = False # valor lógico de falso.
```

```
In [35]: a and b # conjunción lógica
```

```
Out[35]: False
```

```
In [36]: a or b # disyunción lógica
```

```
Out[36]: True
```

```
In [37]: a or True
```

```
Out[37]: True
```

```
In [38]: b and False
```

```
Out[38]: False
```

```
In [39]: a = 2  
b = 4  
b == a * 2 # la comparación devuelve un valor lógico
```

```
Out[39]: True
```

```
In [40]: a * 3 >= 7
```

```
Out[40]: False
```

```
In [41]: b * 2 < 8
```

```
Out[41]: False
```

```
In [42]: b * 2 <= 8
```

```
Out[42]: True
```

```
In [43]: a >= 2 and b < 7
```

```
Out[43]: True
```

```
In [44]: a >= 2 or b < 7
```

```
Out[44]: True
```

```
In [45]: def is_in_circle(x, y, r):  
         return x**2 + y**2 <= r*r  
  
         is_in_circle(0.5, 0.5, 1)
```

```
Out[45]: True
```

```
In [46]: is_in_circle(0.5, 0.9, 1)
```

```
Out[46]: False
```

Cadenas de caracteres

```
In [47]: a = "Hola"  
         b = 'Hola' # las cadenas de caracteres se pueden poner con comillas dobles o simples.  
         a, b
```

```
Out[47]: ('Hola', 'Hola')
```

```
In [48]: a == b
```

```
Out[48]: True
```

```
In [49]: nombre = "Juan"  
         saludo = "Hola " + nombre # + es la concatenación de cadenas de caracteres  
         saludo
```

```
Out[49]: 'Hola Juan'
```

```
In [50]: "H" * 10 # replicación
```

```
Out[50]: 'HHHHHHHHHH'
```

```
In [51]: "Juan" < "Jual" # las cadenas de caracteres se comparan según el orden lexicográfico
```

```
Out[51]: False
```

```
In [52]: "Juan" < "Juap"
```

```
Out[52]: True
```

```
In [53]: "j" == "J" # las mayúsculas y las minúsculas son diferentes
```

```
Out[53]: False
```

```
In [54]: "J" < "j"
```

```
Out[54]: True
```

```
In [55]: len("Hola") # la longitud de una longitud de cadena de caracteres
```

```
Out[55]: 4
```

```
In [56]: len("Adiós") # Las letras con tilde ocupan 2 bytes.
```

```
Out[56]: 6
```

```
In [57]: a = "Adiós"  
a
```

```
Out[57]: 'Adi\xc3\xb3s'
```

```
In [58]: print(a)
```

```
Adiós
```

Cadenas de caracteres largas

```
In [4]: quijote = """En un lugar de la Mancha, de cuyo nombre no quiero acordarme,
no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero,
adarga antigua, rocín flaco y galgo corredor.
Una olla de algo más vaca que carnero, salpicón las más noches,
duelos y quebrantos los sábados, lantejas los viernes, algún palomino de añadidura los
domingos,
consumían las tres partes de su hacienda.
El resto della concluían sayo de velarte, calzas de velludo para las fiestas, con sus
pantuflos de lo mismo,
y los días de entresemana se honraba con su vellorí de lo más fino.
Tenía en su casa una ama que pasaba de los cuarenta, y una sobrina que no llegaba a lo
s veinte,
y un mozo de campo y plaza, que así ensillaba el rocín como tomaba la podadera.
Frisaba la edad de nuestro hidalgo con los cincuenta años; era de complexión recia, se
co de carnes,
enjuto de rostro, gran madrugador y amigo de la caza.
Quieren decir que tenía el sobrenombre de Quijada, o Quesada, que en esto hay alguna d
iferencia en los autores que deste caso escriben;
aunque, por conjeturas verosímiles, se deja entender que se llamaba Quejana.
Pero esto importa poco a nuestro cuento; basta que en la narración dél no se salga un
punto de la verdad."""
```

```
In [5]: print(quijote)
```

```
En un lugar de la Mancha, de cuyo nombre no quiero acordarme,
no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero,
adarga antigua, rocín flaco y galgo corredor.
Una olla de algo más vaca que carnero, salpicón las más noches,
duelos y quebrantos los sábados, lantejas los viernes, algún palomino de añadidura los
domingos,
consumían las tres partes de su hacienda.
El resto della concluían sayo de velarte, calzas de velludo para las fiestas, con sus
pantuflos de lo mismo,
y los días de entresemana se honraba con su vellorí de lo más fino.
Tenía en su casa una ama que pasaba de los cuarenta, y una sobrina que no llegaba a lo
s veinte,
y un mozo de campo y plaza, que así ensillaba el rocín como tomaba la podadera.
Frisaba la edad de nuestro hidalgo con los cincuenta años; era de complexión recia, se
co de carnes,
enjuto de rostro, gran madrugador y amigo de la caza.
Quieren decir que tenía el sobrenombre de Quijada, o Quesada, que en esto hay alguna d
iferencia en los autores que deste caso escriben;
aunque, por conjeturas verosímiles, se deja entender que se llamaba Quejana.
Pero esto importa poco a nuestro cuento; basta que en la narración dél no se salga un
punto de la verdad.
```

Caracteres especiales

```
In [1]: especiall = "Las \"comillas\" son caracteres especiales"
especiall
```

```
Out[1]: 'Las "comillas" son caracteres especiales'
```

In []:

```
In [2]: especial2 = "Las \'comillas\' son caracteres especiales"
especial2
```

Out[2]: "Las 'comillas' son caracteres especiales"

```
In [3]: especial3 = "Por tanto la barra invertida también lo es \\"
especial3
```

Out[3]: 'Por tanto la barra invertida tambi\xc3\xa9n lo es \\'

```
In [10]: especial4 = "Hay otros caracteres especiales como el salto de línea.\nQue sirve para r
representar el fin de línea en ficheros.\nLo veremos más adelante"
print(especial4)
```

```
Hay otros caracteres especiales como el salto de línea.
Que sirve para representar el fin de línea en ficheros.
Lo veremos más adelante
```

Errores comunes

```
In [24]: def media (a, b):
         return (a + b) / 2 # es la división entre enteros. Conviene usar // por compatibil
         idad
media(1, 10)
```

Out[24]: 5

```
In [25]: def media (a, b):
         return (a + b) / 2.0 # es la división entre enteros. Conviene usar // por compatib
         ilidad
media(1, 10)
```

Out[25]: 5.5

```
In [26]: def media (a, b):
         return float(a + b) / 2 # la función float construye un real a partir de un entero
         .
media(1,10)
```

Out[26]: 5.5

```
In [27]: def paradoja(a):
         b = math.sqrt(a) # la raíz cuadrada de 2
         return a == b * b # esto debería ser cierto siempre, no lo es por el error de la r
         epresentación de los reales
paradoja(2)
```

Out[27]: False

```
In [55]: 0.5 == math.cos(math.pi/3) # teóricamente esto también debería ser cierto.
```

```
Out[55]: False
```

```
In [62]: def fibonacci(n):  
    phi = (1 + math.sqrt(5)) / 2  
    return (phi ** n - (1 - phi) ** n)/math.sqrt(5)  
fibonacci(4)
```

```
Out[62]: 3.0000000000000004
```

```
In [64]: def fibonacci(n):  
    phi = (1 + math.sqrt(5)) / 2  
    return int(round((phi ** n - (1 - phi) ** n)/math.sqrt(5)))  
fibonacci(4)
```

```
Out[64]: 3
```