

03-operaciones-con-numeros

May 25, 2020

1 Operaciones matemáticas

Los valores numéricos son el resultado de una serie de operaciones matemáticas.

Operador	Nombre	Descripción	Ejemplo
+	Suma	Suma los valores de tipo de datos numéricos	$5 + 4 = 9$
-	Resta	Resta los valores de tipo de datos numéricos	$7 - 4 = 3$
*	Multiplicación	Multiplica los valores de tipo de datos numéricos	$6 * 5 = 30$
/	División	Devuelve el resultado de la división	$46 / 3 = 15.33333333$
//	División entera	Devuelve solo el resultado de la parte entera de la división	$46 // 3 = 15$
%	Módulo	No hace otra cosa que devolver el residuo de la división entre los dos operandos	$7 \% 2 = 1$
**	Exponente	Calcula el exponente entre valores de tipo de datos numéricos	$3 ** 2 = 9$
-	Negación	Asigna un valor negativo a un tipo de datos numéricos	-7

1.1 Orden de precedencia

1. Exponente (**)
2. Negación (-)
3. Multiplicación (*) - División (/) - División entera (//) - Módulo (%)
4. Suma (+) - Resta (-)

1.2 Ejemplo

```
[1]: # suma  
5 + 4
```

[1]: 9

```
[2]: # resta  
7 - 4
```

[2]: 3

```
[3]: # multiplicación  
6 * 5
```

[3]: 30

```
[4]: # división  
46 / 3
```

[4]: 15.333333333333334

```
[5]: # división entera  
46 // 3
```

[5]: 15

```
[6]: # módulo  
7 % 2
```

[6]: 1

```
[7]: # exponente  
3 ** 2
```

[7]: 9

```
[8]: # orden de precedencia  
3 ** 3 / 56
```

[8]: 0.48214285714285715

```
[9]: # orden de precedencia  
(5 * 56 + 67) / (100 ** -1)
```

[9]: 34700.0

```
[10]: # negación  
-7
```

[10]: -7

11-operadores-asignacion

May 25, 2020

1 Operadores de asignacion

Lo operadores de asignación se utilizan para asignar valores a las variables.

Operador	Descripción	Ejemplo
=	Es el más simple de todos... asigna a la variable del lado izquierdo cualquier variable o resultado del lado derecho	<code>x = 9</code>
+=	Suma a la variable del lado izquierdo el valor del lado derecho	<code>x = 9 x += 10 x ya equivale a 19</code>
-=	Resta a la variable del lado izquierdo el valor del lado derecho	<code>x = 10 x -= 5 x ya equivale a 5</code>
*=	Multiplica a la variable del lado izquierdo el valor del lado derecho	<code>x = 7 x *= 3 x ya equivale a 21</code>
/=	Divide a la variable del lado izquierdo el valor del lado derecho	<code>x = 46 x /= 3 x ya equivale a 15.33333333</code>
%=	Devuelve el residuo de la división a la variable del lado izquierdo el valor del lado derecho	<code>x = 7 x %= 3 x ya equivale a 1</code>
//=	Calcula la división entera a la variable del lado izquierdo el valor del lado derecho.	<code>x = 46 x //= 3 x ya equivale a 15</code>
**=	Calcula el exponente a la variable del lado izquierdo el valor del lado derecho.	<code>x = 3 x **= 3 x ya equivale a 27</code>

1.1 Ejemplo

```
[1]: # = igual
x = 9
x
```

```
[1]: 9
```

```
[2]: # += suma
x = 9
x += 10
x
```

[2]: 19

```
[3]: # -= resta
x = 10
x -= 5
x
```

[3]: 5

```
[4]: # *= multiplicación
x = 7
x *= 3
x
```

[4]: 21

```
[5]: # /= división
x = 46
x /= 3
x
```

[5]: 15.333333333333334

```
[6]: # %= módulo
x = 7
x %= 3
x
```

[6]: 1

```
[7]: # //= división entera
x = 46
x //= 3
x
```

[7]: 15

```
[8]: # **= exponente
x = 3
x **= 3
x
```

[8] : 27

12-operadores-comparacion

May 25, 2020

1 Operadores de Comparación

Los operadores de comparación se utilizan para comparar dos valores.

Operador	Nombre	Descripción	Ejemplo
<code>==</code>	Igual	Evalua que los valores sean iguales para varios tipos de datos.	<code>5 == 6</code> False
<code>!=</code>	No es igual	Evalua si los valores son distintos.	<code>5 != 6</code> True
<code>></code>	Mayor qué	Evalua si el valor del lado izquierdo es mayor que el valor del lado derecho.	<code>5 > 6</code> False
<code><</code>	Menor qué	Evalua si el valor del lado izquierdo es menor que el valor del lado derecho.	<code>5 < 6</code> True
<code>>=</code>	Mayor ó igual a	Evalua si el valor del lado izquierdo es mayor o igual que el valor del lado derecho.	<code>5 >= 6</code> False
<code><=</code>	Menor ó igual a	Evalua si el valor del lado izquierdo es menor o igual que el valor del lado derecho.	<code>5 <= 6</code> True

1.1 Ejemplo

```
[1]: # igual ==  
5 == 6
```

```
[1]: False
```

```
[2]: # no es igual  
5 != 6
```

```
[2]: True
```

```
[3]: # mayor qué  
5 > 6
```

[3]: False

```
[4]: # menor qué  
5 < 6
```

[4]: True

```
[5]: # mayor ó igual a  
5 >= 6
```

[5]: False

```
[6]: # menor ó igual a  
5 <= 6
```

[6]: True

13-operadores-logicos

May 25, 2020

1 Operadores lógicos

Los operadores lógicos se utilizan para combinar declaraciones condicionales.

Operador	Descripción	Ejemplo
and	Devuelve True si ambas afirmaciones son verdaderas	<code>5 < 6 and 7 > 1</code> True
or	Devuelve True si una de las declaraciones es verdadera	<code>5 < 6 and 7 < 1</code> True
not	Invierte el resultado, devuelve False si el resultado es verdadero	<code>not(5 < 6 and 5 == 5)</code> False

1.1 Ejemplo

```
[1]: # and
5 < 6 and 7 > 1
```

[1]: True

```
[2]: # or
5 < 6 or 7 < 1
```

[2]: True

```
[3]: # not
not(5 < 6 and 5 == 5)
```

[3]: False

```
[4]: # not
not(5 > 6 and 5 != 5)
```

[4]: True

14-operadores-identidad

May 25, 2020

1 Operadores de Identidad

Los operadores de identidad se utilizan para comparar los objetos, no si son iguales, sino si en realidad son el mismo objeto, con la misma ubicación de memoria.

Operador	Descripción	Ejemplo
is	Devuelve True si ambas variables son el mismo objeto	<code>x = 5 y = 5 x is y</code> <code>True</code>
is not	Devuelve True si ambas variables no son el mismo objeto	<code>x = 5 y = 5 x is not y</code> <code>True</code>

Se recomienda usar `is` y `is not` solo en list, tuple y dictionary. Para números y cadena de texto es mejor usar los operadores de comparación.

1.1 Ejemplo

```
[1]: # is
x = 5
y = 5
x is y
```

```
[1]: True
```

```
[2]: # is
x = 5
y = 5.9
x is y
```

```
[2]: False
```

```
[3]: # is not
x = 5
y = 5
x is not y
```

```
[3]: False
```

```
[4]: # is not  
x = 5  
y = 5.9  
x is not y
```

[4]: True

```
[5]: lista_1 = ["manzana", "banano"]  
lista_2 = ["manzana", "banano"]  
lista_3 = lista_1
```

```
[6]: # is  
lista_1 is lista_2
```

[6]: False

Devuelve False porque la lista_1 y lista_2 ocupan espacio de memoria diferente.

```
[7]: lista_3 is lista_1
```

[7]: True

Devuelve True porque la lista_1 y lista_3 ocupan espacio de memoria igual.

15-operadores-membresia

May 25, 2020

1 Operadores de Membresia

Los operadores de membresía se utilizan para probar si una secuencia se presenta en un objeto.

Operador	Descripción	Ejemplo
in	Devuelve True si una secuencia con el valor especificado está presente en el objeto	<code>x = [1, 2, 3] 3 in x</code> True
not in	Devuelve True si una secuencia con el valor especificado no está presente en el objeto	<code>x = [1, 2, 3] 2 not in x</code> False

Los operadores de membresia se pueden usar en list, tuple, dictionary o string.

1.1 Ejemplo

```
[1]: # string
fruta = "manzana"
"man" in fruta
```

```
[1]: True
```

```
[2]: # lista
x = [1, 2, 3]
```

```
[3]: # in
3 in x
```

```
[3]: True
```

```
[4]: # not in
2 not in x
```

```
[4]: False
```

```
[5]: lista = ["manzana", "banano", "sandia"]
```

```
[6]: # in  
     "mango" in lista
```

[6]: False

```
[7]: # not in  
     "mango" not in lista
```

[7]: True

20200525

May 25, 2020

```
[1]: # Operadores matematicos  
5 + 4 / 8 * 2
```

[1]: 6.0

```
[2]: 3 ** (3 + 2) - 8
```

[2]: 235

```
[3]: # módulo  
9 % 3
```

[3]: 0

```
[4]: # división  
21 / 5
```

[4]: 4.2

```
[5]: # devolver la parte entera la división  
21 // 5
```

[5]: 4

0.1 Operadores de asignación

```
[6]: x = 56  
# x = x + 4 -> 60  
x += 4  
x
```

[6]: 60

```
[7]: x = 3  
# x = x ** 2 -> 9  
x **= 2  
x
```

[7]: 9

```
[8]: x = 67
      y = 3
      # y = y + x
      y += x
      y
```

[8]: 70

```
[9]: x = -4
      y = 60
      z = 0
      z += x
      print(z)
      z += y
      print(z)
```

-4
56

```
[10]: texto_1 = "Hola"
      texto_1 += " mundo"
      texto_1
```

[10]: 'Hola mundo'

0.2 Operadores de Comparación

```
[11]: 5 == 6
```

[11]: False

```
[12]: 6 < 6
```

[12]: False

```
[13]: 90 >= 90
```

[13]: True

```
[14]: 70 != 70
```

[14]: False

```
[15]: 90 > 35
```

[15]: True

```
[16]: 90 <> 90
```

```
File "<ipython-input-16-c6ca58933099>", line 1
90 <> 90
    ^
SyntaxError: invalid syntax
```

0.3 Operadores lógicos

```
[17]: # and
6 > 5 and 5 < 9
```

[17]: True

```
[18]: # and
7 < 5 and 7 > 2
```

[18]: False

```
[19]: # or
6 > 5 or 5 < 9
```

[19]: True

```
[20]: # or
7 < 5 or 7 > 2
```

[20]: True

```
[21]: # not
not(6 > 5 and 5 < 9)
```

[21]: False

```
[22]: # not
not(7 < 5 and 7 > 2)
```

[22]: True

0.4 Operadores de Identidad

```
[23]: x = 5  
      y = 5  
      x is y
```

[23]: True

```
[24]: x = 5  
      y = 6  
      x is y
```

[24]: False

```
[25]: x = "ana"  
      y = "ana"  
      x is y
```

[25]: True

```
[26]: x = 5  
      y = x  
      x is y
```

[26]: True

```
[27]: y = 6  
      x is y
```

[27]: False

```
[28]: lista_1 = ["manzana", "banano"]  
      lista_2 = ["manzana", "banano"]  
      lista_3 = lista_1
```

```
[29]: lista_1 is lista_2
```

[29]: False

```
[30]: lista_1 is lista_3
```

[30]: True

```
[31]: lista_1 == lista_2
```

[31]: True

```
[32]: lista_1 is not lista_2
```



```
[32]: True
```

0.5 Operadores de Membresia

```
[33]: lista_1 = ["fresa", "manzana", "mandarina", 2]
```

```
[34]: "mango" in lista_1
```

```
[34]: False
```

```
[35]: "fres" in lista_1
```

```
[35]: False
```

```
[36]: "fresa" in lista_1
```

```
[36]: True
```

```
[37]: lista_1.append("naranja")
```

```
[38]: lista_1
```

```
[38]: ['fresa', 'manzana', 'mandarina', 2, 'naranja']
```

```
[39]: 2 not in lista_1
```

```
[39]: False
```

```
[40]: 2 in lista_1
```

```
[40]: True
```

```
[41]: fruta = "manzana"  
      "man" in fruta
```

```
[41]: True
```

```
[42]: for x in lista_1:  
      if "man" in str(x):  
          print(f"Lo encontré: {x}")
```

```
Lo encontré: manzana  
Lo encontré: mandarina
```

```
[43]: x = range(1000000)
```

```
[44]: 7850000 in list(x)
```

[44]: False

[45]: 785000 in list(x)

[45]: True

Debemos ver todos los vídeos de la sección 8 (del 45 al 55) del curso que estamos siguiendo