NIVEL 1

EXPRESIONES & OPERADORES (ARITMÉTICOS Y SOBRE CADENAS)

Universidad de los Andes

EXPRESIONES

- Una expresión es una combinación de valores, variables y operadores
- La evaluación de una expresión produce un valor. Por esto las expresiones pueden aparecer al lado derecho de una asignación:

variable = valor o expresión

Una expresión es diferente a una asignación

```
In [46]: 3+7
Out[46]: 10

In [47]: 8/2
Out[47]: 4.0

In [48]: 7/2
Out[48]: 3.5
```

```
In [52]: x
Out[52]: 10

In [53]: y
Out[53]: 4.0

In [54]: z
Out[54]: 3.5
```

Una expresión se evalúa y devuelve un valor, si este no se asigna a una variable se pierde

```
In [49]: x = 3+7

In [50]: y = 8/2

In [51]: z = 7/2
```

Una asignación es una instrucción que se ejecuta y no devuelve nada (es "muda")

OPERADORES Y OPERANDOS ARITMÉTICOS

- Los operadores son símbolos que representan cálculos como las operaciones aritméticas
- Los valores que usan los operadores son llamados operandos
- Los operandos pueden ser: valores, variables o expresiones

Operadores Aritméticos

Operación	Operador	Aridad	Precedencia
Exponenciación	**	Binario	1
Identidad	+	Unario	2
Cambio de signo	-	Unario	2
Multiplicación	*	Binario	3
División	/	Binario	3
División entera	//	Binario	3
Módulo (o resto)	%	Binario	3
Suma	+	Binario	4
Resta	-	Binario	4

El nivel de precedencia 1 es el de mayor prioridad y el 4 el de menor

Para valores positivos, x // y equivale a la división entera. Para valores negativos devuelve el mayor número entero menor o igual al resultado de la división

EJERCICIO

¿Qué resultados obtendrás al evaluar las siguientes expresiones en Python?

a) 2 + 3 + 1 + 2

b) 2 + 3 * 1 + 2

c) (2 + 3) * 1 + 2

d) (2 + 3) * (1 + 2)

e) +---6

f) -+-+6

g) -3 / 2 - 1

h) -3 // 2 - 1

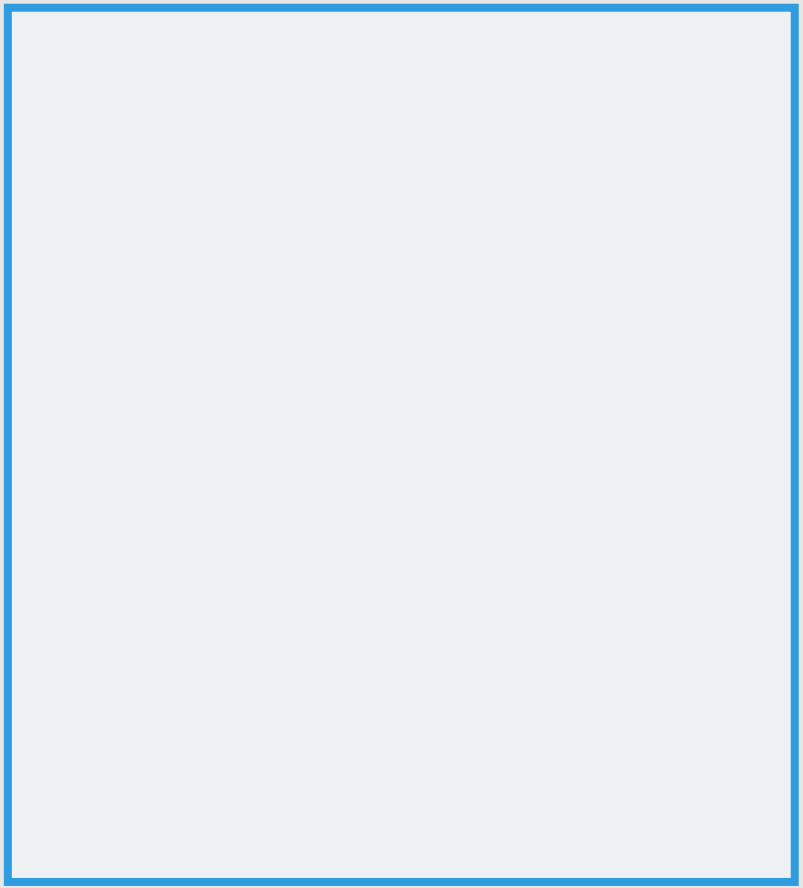
ASIGNACIONES CON OPERADOR

Analicemos la siguiente secuencia de instrucciones:

```
In [30]: i = 10

In [31]: i = i + 1
```

¿Qué valor tiene i aquí?



ASIGNACIONES CON OPERADOR

Analicemos la siguiente secuencia de instrucciones:

```
In [30]: i = 10

In [31]: i = i + 1

In [32]: i
Out[32]: 11
```

Incrementar el valor de una variable en una cantidad cualquiera es tan frecuente en programación que existe una forma compacta:

```
In [33]: i+=1

In [34]: i
Out[34]: 12
```

Todos los operadores aritméticos tienen su asignación con operador asociada:

```
In [41]: z=1

In [42]: z+=2

In [43]: z*=2

In [44]: z//=2

In [45]: z-=2

In [46]: z%=2

In [47]: z**=2

In [48]: z/=2

In [49]: z
Out[49]: 0.5
```

OPERACIONES SOBRE STRINGS

¡En general, NO se pueden efectuar operaciones aritméticas sobre strings!

```
In [51]: nombre = "Pepe"

In [52]: nombre - 1
Traceback (most recent call last):

  File "<ipython-input-52-0433c361dbc6>", line 1, in <module>
    nombre - 1

TypeError: unsupported operand type(s) for -: 'str' and 'int'
```

Pero es posible + "sumar" strings entre sí. Esto se llama concatenar:

```
In [54]: nombre = "Pepe"

In [55]: apellido = "Rojas"

In [56]: nombre_completo = nombre + apellido

In [57]: nombre_completo
Out[57]: 'PepeRojas'

In [58]: nombre = "Pepe "

In [59]: nombre_completo = nombre + apellido

In [60]: nombre_completo
Out[60]: 'Pepe Rojas'
```

OPERACIONES SOBRE STRINGS

También es posible repetir strings. Se denota con el símbolo *, concatenando la cadena consigo misma tantas veces como indique el número entero:

```
In [62]: "Hola" * 5
Out[62]: 'HolaHolaHolaHolaHola'

In [63]: '-' * 60
Out[63]: '------------------------------------------------------------'

In [64]: 60 * '-'
Out[64]: '------------------------------------------------------------'

In [65]: "Hola " * 5
Out[65]: 'Hola Hola Hola Hola Hola '
```

EJERCICIO

¿Qué resultado obtendrás al evaluar la siguiente expresión en Python?

✓ 'a' * 3 + '/*' * 5 + 2 * 'abc' + '+'