



PONTIFICIA  
UNIVERSIDAD  
CATÓLICA  
DE CHILE

## INTRODUCCIÓN A LA PROGRAMACIÓN

---

# Valores, variables, entrada y salida

- Resumen cátedra pasada
  - Python
  - Ejemplos sencillos
- Contenidos de esta cátedra
  - Profundizar sobre tipos
  - Operaciones aritméticas
  - Operaciones lógicas

# Situando Python

- Alto nivel
  - ¿Por qué es de alto nivel y no de bajo nivel?
- Interpretado
  - ¿Qué quiere decir esto?
- Multi-paradigma:
  - Estructurado (este curso)
  - Orientado a objetos
  - Funcional
- Tiene muchas *bibliotecas* (a.k.a. *librerías*)

# Luego programamos un poco

- ¿Qué es una variable? ¿Para qué sirve?
- ¿Qué es un valor?
- ¿Qué es una expresión?
- ¿Qué imprimen los siguientes *print*?

```
print( 1000 * 5 / 10 )
```

```
print( 'hola', "mundo" )
```

```
print( input() )
```



# Códigos de ejemplo de la clase pasada

```
1  # un codigo de ejemplo
2
3  x = 5          # Definimos una variable llamada x; guardamos 5 en ella
4
5  y = 6.5        # Definimos una variable llamada y; guardamos 6.5 ahi
6
7  z = x + y      # La variable z guarda la suma de los valores en x e y
8
9  print( z )     # Imprimimos en pantalla el resultado: es 11.5
10
11 # Notemos que 5 es un numero entero, o sea, tipo int
12 # En cambio, 6.5 es un 'numero de punto flotante', o sea, float
13 # Su suma entrega un numero de punto flotante (float), 11.5
```



# Ojo con los flotantes

```
1  # un codigo de ejemplo
2
3  x = 5                # Definimos una variable llamada x; guardamos 5 en ella
4
5  y = 5.0              # Definimos una variable llamada y; guardamos 5.0 ahi
6
7  z = x + y            # La variable z guarda la suma de los valores en x e y
8
9  print( z )           # Imprimimos en pantalla el resultado: es 10.0
10
11 # Notemos que 5 es un numero entero, o sea, tipo int
12 # En cambio, 5.0 es un 'numero de punto flotante', o sea, float
13 # Lo anterior, pese a que 5 y 5.0 sean matematicamente iguales
14 # Python no hace el esfuerzo de convertir 5.0 en 5
15 # Su suma entrega un numero de punto flotante (float), 10.0
```



# Strings y concatenacion

```
1  # un codigo de ejemplo
2
3  x = "MOTO"          # Guardamos el texto HOLA en la variable x
4
5  y = 'MaMi'          # Guardamos el texto MaMi en la variable y
6
7  z = x + y           # Ambos textos son concatenados (pegados) y guardados
8                      # en la variable z
9
10 print( z )          # Imprimimos en pantalla el resultado: MOTOMaMi
11
12 # Los textos son de tipo str (string, cadenilla)
13 # Los textos se pueden concatenar o pegar (no restar)
14 # Para especificar un texto podemos iniciar y terminar con comillas
15 # simples (') o bien con comillas dobles ("), pero siempre abriendo
16 # y cerrando con el mismo tipo de comillas
```



# Error de tipo

```
1  # un codigo de ejemplo
2
3  x = "MOTO"          # Guardamos el texto HOLA en la variable x
4
5  y = 15.0            # Guardamos el flotante 15.0 en la variable y
6
7  z = x + y           # ERROR: no podemos sumar un texto a un numero
8
9  print( z )          # Esta linea no se ejecuta...
10
11 # EL ERROR:
12 Traceback (most recent call last):
13   File "/home/mauricio/sth.py", line 7, in <module>
14     z = x + y          # ERROR: no podemos sumar un texto a un numero
15         ~~~^~~~
16   TypeError: can only concatenate str (not "float") to str
```





# Multiplicar textos

```
1  # un codigo de ejemplo
2
3  x = "MOTO"          # Guardamos el texto MOTO en la variable x
4
5  y = 5               # Guardamos el entero (int) 5 en la variable y
6
7  z = x * y           # FUNCIONA!!!
8
9  print(z)            # Imprime: MOTOMOTOMOTOMOTOMOTO
10
11 # Si multiplicamos un texto (tipo str) por un numero entero
12 # (tipo int) no negativo, obtenemos un nuevo texto que consiste
13 # en el texto original repetido tantas veces como indica el
14 # numero entero
```



# Pedir entradas

```
1  # un codigo de ejemplo
2
3  print( "Escriba algo:" )
4
5  x = input()      # Pedimos una entrada (al teclado)
6                  # Lo que escribamos se guardara en x
7                  # Sera de tipo texto (str)
8
9  y = input()      # Pedimos otra entrada (al teclado)
10                 # Lo que escribamos se guardara en y
11                 # Sera de tipo texto (str)
12
13 print( "Usted escribio:", x, "--y luego:", y )
14
```

- 1) Pida tres textos (use input) y, luego, imprímalos en el mismo orden solicitado (imprima uno por línea)
- 2) Pida tres textos (use input) e imprímalos en el mismo inverso al solicitado (primero imprima el último, luego el del medio, luego el primero)
- 3) Pida dos textos, concaténelos e imprima su concatenación en pantalla

# Operaciones

# Operadores de asignación

## OPERADOR

## DESCRIPCIÓN

=

$a = 5$ . El valor 5 es asignado a la variable  $a$

+=

$a += 5$  es equivalente a  $a = a + 5$

-=

$a -= 5$  es equivalente a  $a = a - 5$

\*=

$a *= 3$  es equivalente a  $a = a * 3$

/=

$a /= 3$  es equivalente a  $a = a / 3$

%=

$a \% = 3$  es equivalente a  $a = a \% 3$

\*\*=

$a ** = 3$  es equivalente a  $a = a ** 3$

# Operadores aritméticos

OPERADOR	DESCRIPCIÓN	USO
+	Realiza Adición entre los operandos	$12 + 3 = 15$
-	Realiza Substracción entre los operandos	$12 - 3 = 9$
*	Realiza Multiplicación entre los operandos	$12 * 3 = 36$
/	Realiza División entre los operandos	$12 / 3 = 4$
%	Realiza un módulo entre los operandos	$16 \% 3 = 1$
**	Realiza la potencia de los operandos	$12 ** 3 = 1728$
//	Realiza la división con resultado de número entero	$18 // 5 = 3$

# Operadores de comparación

## OPERADOR

## DESCRIPCIÓN

## USO

>

Devuelve True si el operador de la izquierda es mayor que el operador de la derecha

12 > 3 devuelve True

<

True si el operador de la derecha es mayor que el operador de la izquierda

12 < 3 devuelve False

==

True si ambos operandos son iguales

12 == 3 devuelve  
False

>=

True si el operador de la izquierda es mayor o igual que el operador de la derecha

12 >= 3 devuelve True

<=

True si el operador de la derecha es mayor o igual que el operador de la izquierda

12 <= 3 devuelve  
False

!=

True si ambos operandos no son iguales

Cierre