### ACÁMICA

#### **Herramientas**

## ¿Tienen acceso?







# Auto-reflexión ¿Pudieron hacerla?

#### Te sugerimos que hagas todo el Coding route

Es un recurso complementario que te ayudará a reforzar tus conocimientos en programación.

Cada concepto teórico tiene un ejercicio práctico asociado. Los conceptos son incrementales por lo que te sugerimos que inicies por el capitulo 1 *Introducción a la programación*.

Iniciar mi Coding route

#### ¡Comienzas con una buena base!

Cuentas con los conocimientos y habilidades necesarias para cursar el programa intensivo de Data Science.

Si quieres sacar el mayor provecho a tus estudios, accede a tu code route en dónde encontrarás recursos complementarios para ir sumergiéndote en algunas de las temáticas que surgirán desde el comienzo.

Explorar mi Coding route



### **Python**

¡Continuamos aprendiendo a programar! Hoy vamos a ver dos herramientas sumamente útiles, Listas y Loops.



#### **Agenda**

Daily

Explicación: Listas y Loops.

Break.

Hands-on training: Listas y Loops

Cierre.



## **Daily**





#### **Daily**

#### Sincronizando...

#### **Toolbox**



¿Cómo te ha ido? ¿Obstáculos? ¿Cómo seguimos?

#### Challenge



¿Cómo te ha ido? ¿Obstáculos? ¿Cómo seguimos?



### Primeros pasos en Python





#### UN LENGUAJE DE PROGRAMACIÓN VIENE CON...

tipos de datos básicos Números, texto, variables de verdad (bool), etc.

estructuras de datos

Podemos hacer "conjuntos" de cosas y agruparlas de formas específicas. ¡Y vienen con funcionalidades propias! Ejemplo: listas.

funciones propias

Ejemplo: print(), type(), etc.



Vimos, además, que podemos definir Variables.

#### **VARIABLES**

En un lenguaje de programación, a los datos se los guarda en forma de variables. A cada variable debemos darle un nombre único que la identifique:

```
In [ ]: a = 5
In [ ]: un_nombre_cualquiera = 12.7
In [ ]: b = 'Hola!'
In [ ]: nueva_variable = True
```

A estas **variables** pueden se le pueden asignar distintos **tipos de datos**.

#### VARIABLES y **TIPOS DE DATOS**

Python identifica automáticamente el **tipo de dato** de cada variable. Esto resulta muy cómodo para trabajar.

Pero debemos ser cuidadosos, **a veces** el tipo asignado automáticamente **no es el que esperamos ...** 

#### **TIPOS DE DATOS**

¿Podemos pasar de un **tipo de dato a otro?** 

¡Sí! La **solución** es ser explícitos si deseamos que nuestra variable sea de algún tipo en particular.





#### **TIPOS DE DATOS**

Enteros	Floats	Strings	Booleanos
Son los números que usamos para contar, el 0 y los negativos	Son los números "con coma" Se introducen usando puntos	Texto  Se introducen entre comillas dobles, "", o simples, ".	Variables de "verdad": verdadero o Falso
-1 0 1 2	5.1 -1.3 1.0 10.0	"Hola Mundo" "A" 'Mi nombre es Esteban'	True False 1 == 2 1 == 1
[1]: type(3) [1]: int	[1]: type(3.0) [1]: float	[1]: type("Hola") [1]: str	[1]: type(2==2) [1]: bool

#### Operaciones básicas entre ENTEROS y FLOATS

In [42]: 
$$x = 3$$
  
 $y = 1.5$   
print(x/y)  
2.0

In [43]: 
$$x = 2$$
  
 $y = 3$   
print(x\*\*y)

Operación	Operador	Ejemplo
Suma	+	3 + 5.5 = 8.5
Resta	2	4 - 1 = 3
Multiplicación	*	3 * 6 = 18
Potencia	**	3 ** 2 = 9
División (cociente)	/	15.0 / 2.0 = 7.5
División (parte entera)	//	15.0 // 2.0 = 7
División (resto)	%	7 % 2 = 1



### **Listas y Loops**





## Listas y Loops





#### **Definición**

Una estructura de dato muy importante en Python son las **listas**. Una lista consiste en una serie de elementos ordenados:

Las listas se definen con corchetes []

#### **Operaciones con LISTAS**

Las listas se pueden **sumar** entre sí (se **concatenan**). También se les puede agregar un elemento nuevo mediante el método '**.append()**'

```
In [52]: lista_1 = [2, 4.7, True, 'Texto']
    lista_2 = [42, 42]
    lista_1 + lista_2

Out[52]: [2, 4.7, True, 'Texto', 42, 42]

In [53]: lista_1 = [2, 4.7, True, 'Texto']
    lista_1.append('Un nuevo elemento')
    lista_1

Out[53]: [2, 4.7, True, 'Texto', 'Un nuevo elemento']
```

#### **Operaciones con LISTAS**

```
In [55]: lista_1 = [2, 4.7, True, 'Texto']
len(lista_1)

Out[55]: 4

In [56]: lista_2 = [0, lista_1, 'Mas texto']
len(lista_2)

Out[56]: 3
```

Las listas tienen un largo determinado por su cantidad de elementos. Se consulta mediante la función **len()**.

```
In [59]: lista_vacia = []
len(lista_vacia)

Out[59]: 0

In [60]: lista_vacia.append(42)
lista_vacia.append('un segundo item')
print(lista_vacia)

[42, 'un segundo item']
```

Se pueden generar listas vacías y luego ir agregándole elementos a medida que una lo precise.

### **Listas y Loops**





## Listas y Loops

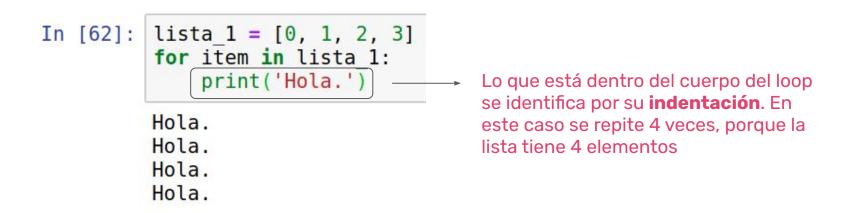




#### LOOPS - For

Los **Loops** en programación son bloques de código que, dadas ciertas condiciones, se repiten una cierta cantidad de veces.

El **For** es un tipo de **Loop** que repite un bloque de código tantas veces como elementos haya en una **lista** dada:



#### LOOPS - For

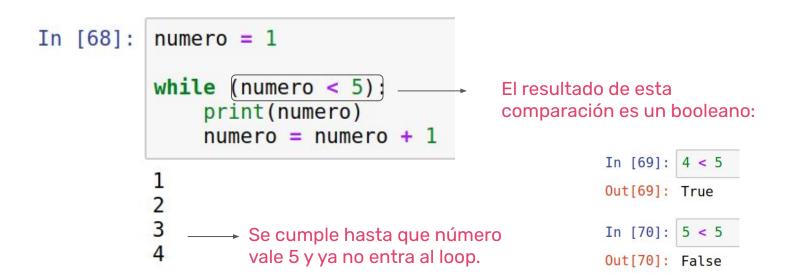
En cada repetición, la variable **item** (podría tener cualquier nombre) va tomando el valor de cada un de los elementos de la lista dada.

#### LOOPS - For

Las listas pueden contener texto. Veamos un ejemplo donde creamos una nueva lista.

#### **LOOPS - While**

El **While** es un tipo de **Loop** que repite un bloque de código hasta que una dada condición se deje de cumplir. Esta condición debe expresarse como una variable **Booleana**.





### **Hands-on training**





Hands-on training



### Trabajamos en el Notebook que descargaste en la Toolbox 02, Sección 2: Listas y Loops

### Recursos





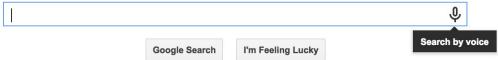
#### **Python**

- https://learnxinyminutes.com/docs/python3/ Exclusivo sobre programación en Python, sin mucho contexto y directo al grano. Minimalista, pero detallado.
- <u>https://www.tutorialsteacher.com/python</u> Muy completo. Útil para "tener a mano".



#### Recomendaciones

- 1) Comentar el código en voz alta ayuda a aprender y a entender lo que estás haciendo.
- 2) No tengas miedo de hacer, romper y arreglar.
- 3) La frustración es una buena señal ("Get things done").
- 4) Pedir la opinión de tus compañeros/as y Squad Leads/as sobre tu código.
- 5) Busca crecer en comunidad (Medium, Github, Slack Stackoverflow, etc).
- 6) Pide ayuda a tu mejor amigo:





#### Data Science

Sprint 1 (Meeting #2)

Nos tomamos unos minutos para completar <u>esta encuesta</u>.

Queremos saber cómo valoran mi tarea hasta acá. ¡Les va a llevar solo un minuto!



### Para la próxima

- Termina el notebook de hoy
- Lee la Toolbox 03
- Resuelve el Challenge

### ACAMICA