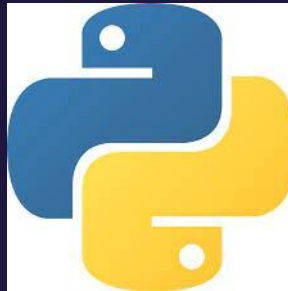


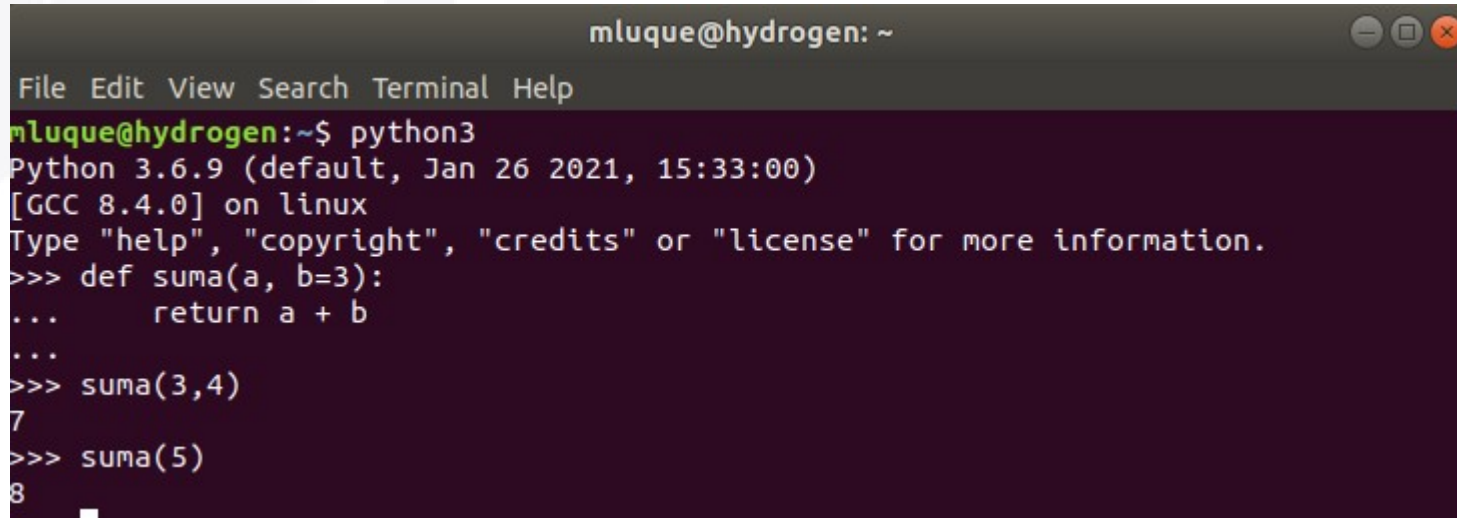
# LENGUAJES y HERRAMIENTA PARA CIENCIAS DE DATOS I

**Funciones II**  
**Argumentos indeterminados**  
**Return varios valores**



# Argumentos por defecto

- Definir valor por defecto en la definición

A terminal window titled 'mluque@hydrogen: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the execution of 'python3', displaying the Python version (3.6.9), GCC version (8.4.0), and OS (linux). It then shows a Python function definition 'def suma(a, b=3):' with a return statement 'return a + b'. The function is called twice: 'suma(3,4)' returns 7, and 'suma(5)' returns 8.

```
mluque@hydrogen: ~  
File Edit View Search Terminal Help  
mluque@hydrogen:~$ python3  
Python 3.6.9 (default, Jan 26 2021, 15:33:00)  
[GCC 8.4.0] on linux  
Type "help", "copyright", "credits" or "license" for more information.  
>>> def suma(a, b=3):  
...     return a + b  
...  
>>> suma(3,4)  
7  
>>> suma(5)  
8
```

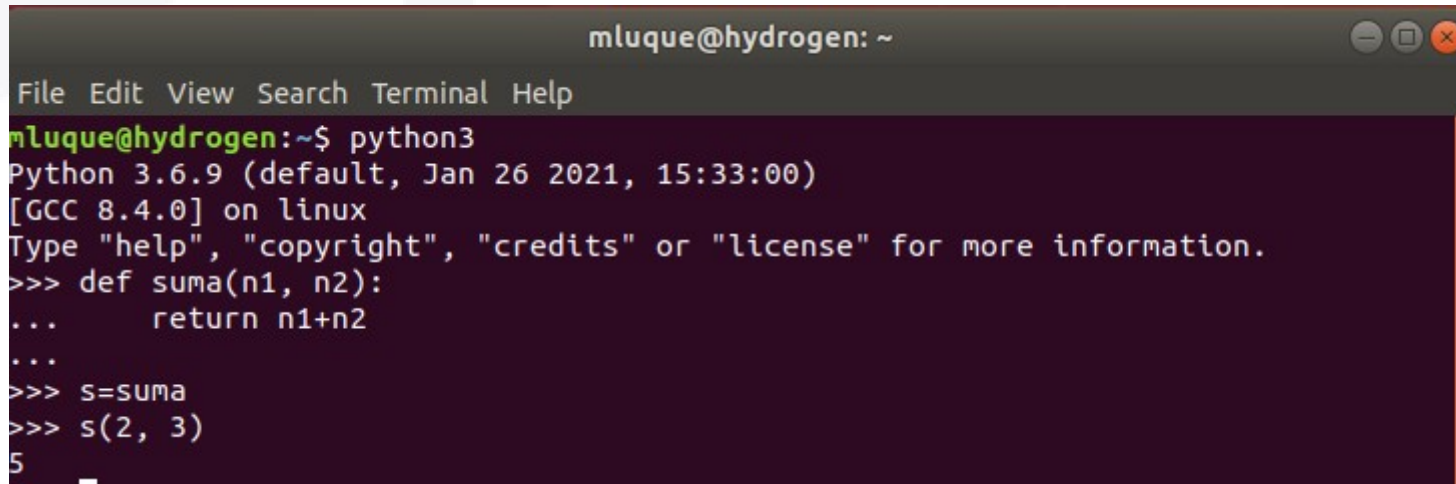
# Palabras claves como argumento

- Alterar el orden de los parámetros indicando el nombre

```
>>> suma(b=2,a=7)
9
>>> suma(b=2,7)
File "<stdin>", line 1
SyntaxError: positional argument follows keyword argument
>>> suma(7,c=2)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: suma() got an unexpected keyword argument 'c'
>>> █
```

# Funciones como argumentos

- Una función puede asignarse a una variable
  - ◆ Llamar a la función a través de la variable



```
m luque@hydrogen: ~  
File Edit View Search Terminal Help  
m luque@hydrogen:~$ python3  
Python 3.6.9 (default, Jan 26 2021, 15:33:00)  
[GCC 8.4.0] on linux  
Type "help", "copyright", "credits" or "license" for more information.  
>>> def suma(n1, n2):  
...     return n1+n2  
...  
>>> s=suma  
>>> s(2, 3)  
5
```

# Funciones como argumentos

```
mluque@hydrogen: ~  
File Edit View Search Terminal Help  
mluque@hydrogen:~$ python3  
Python 3.6.9 (default, Jan 26 2021, 15:33:00)  
[GCC 8.4.0] on linux  
Type "help", "copyright", "credits" or "license" for more information.  
>>> def cuadrado(num):  
...     return num*num  
...  
>>> def aplica(lista, operacion):  
...     for i in lista:  
...         print(operacion(i))  
...  
>>> aplica([1,2,3], cuadrado)  
1  
4  
9
```

# Argumentos especiales

```
def funcion(arg_solo_posicionales, /, posicionales_clave, *, arg_solo_clave)
```

num, 2, lista

num  
clave1=2  
lista

clave1=num  
clave2=2  
clave3=lista

# Argumentos arbitrarios

- Número indefinido de argumentos
  - ◆ \*argumentos

```
File Edit View Search Terminal Help
mluque@hydrogen:~$ python3
Python 3.6.9 (default, Jan 26 2021, 15:33:00)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> def imprimir_parametros(parametro_fijo, *parametros_indefinidos):
...     print(f'Fijo:{parametro_fijo}')
...     for i in parametros_indefinidos:
...         print(i)
...
>>> imprimir_parametros(2, 1, 3, 5)
Fijo:2
1
3
5
```

# Desempaquetado de argumentos

- Situación inversa a la anterior
  - ◆ La función recibe x argumentos fijos
  - ◆ Al invocar la función están almacenados en una lista
    - \*lista

```
mluque@hydrogen:~$ python3
Python 3.6.9 (default, Jan 26 2021, 15:33:00)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> def calcular(precio, descuento):
...     return precio - (precio * descuento / 100)
...
>>> datos=[100, 50]
>>> print(calcular(*datos))
50.0
_
```



# Return con varios valores

- Separarlos por coma

```
File Edit View Search Terminal Help
mluque@hydrogen:~$ python3
Python 3.6.9 (default, Jan 26 2021, 15:33:00)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> def division(dividendo, divisor):
...     cociente = dividendo / divisor
...     resto = dividendo % divisor
...     return cociente, resto
...
>>> print(division(7, 3))
(2.3333333333333335, 1)
>>> c, r = division(7, 3)
>>> print(c, r)
2.3333333333333335 1
>>> d = division(7, 3)
>>> d
(2.3333333333333335, 1)
```

