TIC

Módulo 1

LECCIÓN 2

Funciones propias

y entonces, ¿Qué es una función?

Una función no es más que un envoltorio para un grupo de instrucciones que realizan una o varias acciones. Esas instrucciones pueden esperar valores externos, los cuales se les pueden pasar a través del envoltorio, estos datos son conocidos como parámetros.

Una función solo ejecuta hasta que es invocada, para invicaruna función debe utilizar si nombre y entre los paréntesis, si aplica, agregar cada uno de los argumentos que se requieren para ejecutarla.

Una función que espere 2 parámetros pero al momento de invocarla se le envíen más o menos argumentos arrojará un error.

```
[ ]
    def operacion(a, b):
      """
      Descripción de la función. Como debe ser usada,
        que parámetros acepta y que devuelve
      """
      print(isinstance(a, int) or isinstance(a, float))
      if(a > b):
        return (a - b)
      elif(a < b):
        return (b - a)
      else:
        return (a * b)


    operacion(3.9,5)

    help(operacion)
```

```
    True
    Help on function operacion in module __main__:

    operacion(a, b)
        Descripción de la función. Como debe ser usada,
          que parámetros acepta y que devuelve
```

Pasando argumentos

Las funciones pueden ejecutarse, de forma base, sin recibir argumentos, recibiendo argumentos por posición, argumentos por nombre y argumentos por defecto.

```
# Sin argumentos
def saludar():
  print("hola");
```

Los argumentos por posición o posicionales son la forma más básica e intuitiva de pasar parámetros. Si tenemos una función que acepta dos parámetros, se debe llamar con los dos parámetros.

```
# Argumentos por posición
def saludar(nombre):
  print("hola" + nombre);

# Ejecuta está función, qué sucede?
def sumar(a, b):
  print(a + b)

sumar(5)
sumar(5, 4, 3)
```

Otra forma de llamar a una función, es usando el nombre del argumento con = y su valor. El siguiente código hace lo mismo que el código anterior, con la diferencia de que losargumentos no son posicionales.

```
# Argumentos por nombre
sumar(b = 5, a = 8)
```

Tal vez queramos tener una función con algún parámetro opcional, que pueda ser usado o no dependiendo de diferentes circunstancias. Para ello, lo que podemos hacer es asignar un valor por defecto a la función. En el siguiente caso c valdría cero salvo que se indique lo contrario.

```
def suma(a, b, c=0):
    return a+b+c
suma(5,5,3) # 13
```

Dado que el parámetro c tiene un valor por defecto, la función puede ser llamada sin ese valor.

```
suma(4,3) # 7
```