# **Funciones integradas.**

Las funciones son codigo que ya fue creado como la función print, input, etc. Y las podemos reutilizar en cualquier momento de nuestro código, esto ayuda a evitar repetir codigo muchas veces, ya que haciendo esto solamente tenemos que hacer referencia a la función, ayuda mejorar el modularidad de nuestro codigo, y facilita muchísimo la legibilidad de nuestro codigo.

Tenemos dos tipos de funciones, las que son creadas por Python o tambien llamadas funciones build-in y las que podemos crear nosotros mismos.

Ejemplos de funciones build-in son:

*#Encontrando el numero mayor de una lista*

numeros = [4, 7, 2, 9, 10]

print(max(numeros))

Y esto lo que hace es devolvernos el numero mas alto de en este caso nuestra lista o sea 10, gracias a la función “max”, encuentra el numero más alto de un iterable.

*#Encontrando el numero menor de un iterable*

numeros = [4, 7, 2, 9, 10]

print(min(numeros))

Aquí hace lo contrario o sea devuelve el numero mas pequeño de un elemento iterable gracias a la función “min”.

*#redondeando a seis decimales*

numero = round(12.3567326)

print(numero)

Devuelve un numero redondeado hacia abajo, por lo tanto en este ejemplo nos devuelve 12. Y nos da la oportunidad de elegir cuantos decimales queremos de esta manera.

*#redondeando a seis decimales*

numero = round(12.345678,2)

print(numero)

Con una coma seguida del numero de decimales que queremos, en este caso elegimos dos decimales y que se redondee, entonces nos devuelve, “12.35”.

*#devuelve True si todos los valores son verdaderos*

resultado\_all = all([23, "True", [123,423]])

print(resultado\_all)

Haciendo esto nos devuelve True ya que todos los elementos dentro de la variable “resultado\_all” son verdaderos. Pero si añadimos valores falsos devolverá false.

*#devuelve True si todos los valores son verdaderos*

resultado\_all = all([0 , 23, "True", [123,423, None]])

print(resultado\_all)

En este caso el 0 y None son elementos que son falsos, o sea por lo tanto nos devolverá False con la funciona all.

*#Suma todos los valores de un iterable*

resultado\_sum = sum([1, 2, 3, 4, 5])

print(resultado\_sum)

Entonces nos suma todos los elementos de nuestra lista en este caso. Solo números porque si ponemos una cadena nos devolverá una excepción

Existen muchas mas funciones build-in, para investigar mas buscar en las documentaciones oficiales.