Lista de funciones built-in de Python

Función	Descripción	Ejemplos de uso
int( )	Convierte al tipo de dato int	var_1 = int(‘10’) # var_1 es el número entero 10
float( )	Convierte al tipo de dato float	var_1 = float(‘10’) # var_1 es el número float 10.0
str( )	Convierte al tipo de dato string	var_1 = str(10) var_2 = str(10.0) # var_1 es el string ‘10’ # var_2 es el string ‘10.0’
bool( )	Convierte al tipo de dato boolean	var_1 = bool(1) var_2 = bool(0) # var_1 es el booleano True # var_2 es el booleano False
list( )	Convierte al tipo de dato list	var_1 = list(‘abc’) # var_1 es [‘a’, ‘b’, ‘c’]
tuple( )	Convierte al tipo de dato tuple	var_1 = tuple(‘abc’) # var_1 es (‘a’, ‘b’, ‘c’)

type( )	Retorna un valor tipo “type” indicando el tipo de dato	var_1 = type(10) # var_1 es un type int
		var_2 = type(10.0) # var_2 es un type ‘float’
		var_3 = type(‘10’) # var_3 es un type ‘str’
		var_4 = type(10 > 0) # var_4 es un type ‘bool’
		var_5 = type([‘a’, ‘b’, ‘c’]) # var_5 es un type ‘list’
		var_6 = type((‘a’, ‘b’, ‘c’)) # var_6 es un type ‘tuple’

print( )	Imprime en consola	print(‘Hola’) # se imprime en la consola ‘Hola’
input( )	Imprime en consola y aguarda que el usuario ingrese un dato	var_1 = input(‘Ingrese un numero’) # se imprime en la consola ‘Ingrese un numero’ y aguarda a que el usuario ingrese datos y aprete enter # var_1 es un string con lo que sea que sea haya ingresado el usuario
abs( )	Retorna el valor absoluto de un número	var_1 = abs(-10) # var_1 es el número 10

min( )	Retorna el número más pequeño	var_1 = min([2,4,-5,-20,50]) # var_1 es igual a -20
max( )	Retorna el número más grande	var_1 = min([2,4,-5,-20,50]) # var_1 es igual a 50
len( )	Retorna el largo de una secuencia	var_1 = len([1,4,6,8,10]) # var_1 es igual a 5
range(start, stop, step)	Retorna una secuencia de números enteros equiespaciados, desde start, hasta stop (sin incluirlo) y en pasos step	var_1 = range(1, 10, 2) # la secuencia de números 1,3,5,7,9

any( )	Devuelve True con tal que haya un elemento que sea True. Caso contrario devuelve False	lista1 = [True, 1, -0.29, 'hola']
		lista2 = [False, 1, -0.29, 'hola']
		lista3 = [False, 0, 0.0, '']
		
		bool1 = any(lista1) # bool1 es True
		bool2 = any(lista2) # bool2 es True porque considera que el 1, el -0.29 y el 'hola' son True
		bool3 = any(lista3) # bool3 es False

all( )	Devuelve True sólo si todos los elementos son True. Caso contrario devuelve False	lista1 = [True, 1, -0.29, 'hola']
		lista2 = [False, 1, -0.29, 'hola']
		lista3 = [False, 0, 0.0, '']
		
		bool1 = any(lista1) # bool1 es True
		bool2 = any(lista2) # bool2 es False
		bool3 = any(lista3) # bool3 es False

sequence.count( )	Retorna la cantidad de ocurrencias de un elemento en una secuencia	lista_1 = ['a', 'b', 'a', 'c', 'b', 'a'] var_1 = lista1.count('a') # var_1 es igual a 3
sequence.index( )	Retorna el índice de la primer ocurrencia de un elemento en una secuencia	lista_1 = ['a', 'b', 'a', 'c', 'b', 'a'] var_1 = lista1.index('b') # var_1 es igual a 1
list.append( )	Agrega un elemento al final de una lista	lista_1 = ['a', 'b', 'a', 'c', 'b', 'a'] lista_1.append('d') # lista_1 es ['a', 'b', 'a', 'c', 'b', 'a', 'd']
list.remove( )	Elimina la primer ocurrencia de un elemento en una lista	lista_1 = ['a', 'b', 'a', 'c', 'b', 'a'] lista_1.append('b') # lista_1 es ['a', 'a', 'c', 'b', 'a']
list.insert(index, element)	Inserta un elemento en una posición específica de una lista	lista_1 = ['a', 'b', 'a', 'c', 'b', 'a'] lista_1.insert(3, 'd') # lista_1 es ['a', 'b', 'a', 'd', 'c', 'b', 'a']

list.pop( )	Elimina un elemento de una lista en una posición específica, y retorna dicho elemento	lista_1 = [‘a’, ‘b’, ‘a’, ‘c’, ‘b’, ‘a’] var_1 = lista_1.pop(3) # var_1 es ‘c’ # lista_1 es [‘a’, ‘b’, ‘a’, ‘b’, ‘a’]
list.extend( )	Extiende una lista con los elementos de otra lista	lista_1 = [‘a’, ‘b’, ‘a’, ‘c’, ‘b’, ‘a’] lista_2 = [‘c’, ‘d’, ‘e’] var_1 = lista_1.extend(lista_2) # var_1 es [‘a’, ‘b’, ‘a’, ‘c’, ‘b’, ‘a’, ‘c’, ‘d’, ‘e’]
list.sort( )	Ordena la lista en base a sus contenidos	lista_1 = [3, 2, 1, 5, 6, 9] lista_1.sort( ) # lista_1 es ahora [1, 2, 3, 5, 6, 9]
list.copy( )	Retorna la copia de una lista	lista_1 = [‘a’, ‘b’, ‘a’, ‘c’, ‘b’, ‘a’] lista_2 = lista_1.copy( ) # lista_2 es [‘a’, ‘b’, ‘a’, ‘c’, ‘b’, ‘a’]. Si se cambia un elemento de lista_1, lista_2 no cambia
str.find( )	Retorna el índice de la primer ocurrencia de string	string_1 = ‘Hola’ var_1 = string_1.find(‘o’) # var_1 es igual a 1
str.split( )	Retorna una lista con los elementos siendo los strings que hay entre los separadores	string_1 = ‘Hola%como%estas?’ var_1 = string_1.split(‘%’) # var_1 es igual a [‘Hola’, ‘como’, ‘estas?’]
str.join( )	Retorna un string que es el resultado de concatenar los strings de una lista, usando otro string como “pegamento”	lista_1 = [‘Hola’, ‘como’, ‘estas?’] str_1 = ‘%‘ var_1 = str_1.join(lista_1) # var_1 es igual a ‘Hola%como%estas?’
str.upper( )	Retorna un string con todas las letras mayúsculas	string_1 = ‘Hola como estas?” var_1 = string_1.upper( ) # var_1 es igual a “HOLA COMO ESTAS?”
str.lower( )	Retorna un string con todas las letras minúsculas	string_1 = ‘Hola Como Estas?” var_1 = string_1.upper( ) # var_1 es igual a “hola como estas?”
str.title( )	Retorna un string con la primer letra de cada palabra mayúscula	string_1 = ‘Hola como estas?” var_1 = string_1.upper( ) # var_1 es igual a “Hola Como Estas?”
str.isalpha( )	Retorna un booleano indicando si el string contiene únicamente letras del abecedario	string_1 = ‘Roberto” var_1 = string_1.isalpha( ) # var_1 es igual a True string_2 = ‘Roberto1984”

		var_2 = string_2.isalpha( ) # var_2 es igual a False string_3 = ‘Roberto1984?” var_3 = string_3.isalpha( ) # var_3 es igual a False
str.isalnum( )	Retorna un booleano indicando si el string contiene únicamente letras del abecedario y/o números	string_1 = ‘Roberto” var_1 = string_1.isalnum( ) # var_1 es igual a True string_2 = ‘Roberto1984’ var_2 = string_2.isalnum( ) # var_2 es igual a True string_3 = ‘Roberto1984?” var_3 = string_3.isalnum( ) # var_3 es igual a False
str.isdecimal( )	Retorna un booleano indicando si el string contiene únicamente números	string_1 = ‘Roberto” var_1 = string_1.isdecimal( ) # var_1 es igual a False string_2 = ‘1984’ var_2 = string_2.isdecimal( ) # var_2 es igual a True
str.isupper( )	Retorna un booleano indicando si todas las letras de un string son mayúsculas	string_1 = ‘ROBERTO’ var_1 = string_1.isupper( ) # var_1 es igual a True string_2 = ‘Roberto’ var_2 = string_2.isupper( ) # var_2 es igual a False
str.islower( )	Retorna un booleano indicando si todas las letras de un string son minúsculas	string_1 = ‘roberto’ var_1 = string_1.islower( ) # var_1 es igual a True string_2 = ‘Roberto’ var_2 = string_2.islower( ) # var_2 es igual a False
dict.clear( )	Remueve todos los ítems del diccionario	dict_1 = {‘Pedro’:7 , ‘Lucía’:6 , ‘Paloma’: 10} dict_1.clear( ) # dict_1 está vacio ahora
dict.items( )	Devuelve un objeto iterable con las claves y los valores del diccionario de la forma (clave, valor). Esto es muy	dict_1 = {‘Pedro’:7 , ‘Lucía’:6 , ‘Paloma’: 10} iterable = dict_1.items( ) # iterable guarda un dato de tipo dict_items de la

	útil para ciclos for.	forma: dict_items([('Pedro', 7), ('Lucía', 6), ('Paloma', 10)])
dict.keys( )	Devuelve un objeto iterable con las claves del diccionario. Esto es muy útil para los ciclos for	dict_1 = {‘Pedro’:7 , ‘Lucía’:6 , ‘Paloma’: 10} iterable = dict_1.items( ) # iterable guarda un dato de tipo dict_keys de la forma: dict_keys(['Pedro', 'Lucía', 'Paloma'])
dict.pop( )	Remueve y entrega la clave y su valor del diccionario	dict_1 = {‘Pedro’:7 , ‘Lucía’:6 , ‘Paloma’: 10} valor = dict_1.pop(‘Paloma’) # valor ahora guarda: 10 # dict_1 es ahora : {‘Pedro’:7 , ‘Lucía’:6}