Built-in Functions

- There are several **functions that are readily available for use by the Python interpreter**. These functions are called built-in functions.
- These functions that can be called in the program as and when required, thus saves programmer's time of creating those commonly used functions.
- Categorization of some of the built-in functions are enlisted as:

Input or Output Functions	Datatype Conversion Functions	Mathematical Functions	Random Functions	Other Functions

input()
output()
	int()
	float()
	list()
	set()
	tuple()
	dict()
	str()
		ceil()
		floor()
		abs()
		pow()
		sqrt()
		factorial()
			seed()
			getstate()
			setstate()
				len()
				range()
				find()
				type()

1) Math Functions

- The Python math module provides the mathematical functions to solve mathematical problems.
- To use these functions, it is required to *import math* module.
- Some of the commonly used math functions in *math module* are enlisted in the Table below:

Built-in Math Functions

S. No.	Syntax	Arguments	Returns	Example

1.	math.ceil(x)	x may be an integer or floating point number	ceiling value of x	>>> math.ceil(-9.7)
				-9
				>>> math.ceil (9.7)
				10
				>>> math.ceil(9)
				9

2.	math.floor(x)	x may be an integer or floating point number	floor value of x	>>> math.floor(-4.5)
				-5
				>>> math.floor(4.5)
				4
				>>> math.floor(4)
				4

3.	math.abs(x)	x may be an integer or floating point number	absolute value of x	>>> math.fabs(6.7)
				6.7
				>>> math.fabs(-6.7)
				6.7
				>>> math.fabs(-4)
				4.0

4.	math.factorial(x)	x is a positive integer	factorial of x	>>> math.factorial(5) 120
5.	math.gcd(x,y)	x, y are positive integers	gcd (greatest common	>>> math.gcd(10,2)

			divisor) of x and y	2

6.	math.pow(x,y)	x, y may be an integer or floating point number	xy (x raised to the power y)	>>> math.pow(3,2)
				9.0
				>>> math.pow(4,2.5)
				32.0

7.	math.sqrt(x)	x may be a positive integer or floating point number	square root of x	>>> math.sqrt(144)
				12.0
				>>> math.sqrt(.64)
				0.8

8.	math.sin(x)	x may be an integer or floating point number in radians	sine of x in radians	>>> math.sin(0)
				0
				>>> math.sin(6)
				-0.279

9.	math.pi	No arguments	The value of π	>>>x=math.pi >>>print(x) 3.14

2) Random Functions

- Random functions that are used for generating random numbers.
- To use these functions, it is required to *import random* module.
- Some of the commonly used random functions in *random module* are enlisted in the Table below:

Built-in Random Functions

S. No.	Syntax	Arguments	Example
1.	random.random()	No argument (void)	>>> random.random() 0.65333522
2.	random.seed()	A value to initialize the random number generator	>>> random.seed(10) >>> print(random.random())
3.	random.getstate()	Finds the current state of random number.	>>> print(random.getstate())

4.	random.setstate	State is the current state	>>> state = random.getstate()
			>>> print(random.random())
			>>> random.setstate(state)

5.	random. randrange(x,y)	x and y are positive integers signifying the start and stop value	>>> random.randrange(2,7) 5
6.	random.choice()	list	>>> mylist = ["A", "B"] >>>print(random.choice(mylist)) B

13) Built-in Functions of Built-in Datatypes

Built-in String Functions			
S. No.	**Functions**	**Description**	**Example**
1.	len()	To get the length of a string	>>>a = "Hello, World!" >>>print(len(a)) 13
2.	in keyword	To check if a certain phrase or character is present in a string	>>>txt = "The best things in life are free! >>>print("free" in txt)
3.	upper()	returns the string in upper case	>>>a = "Hello, World!" >>>print(a.upper())
4.	lower()	returns the string in lower case	>>> a = "Hello, World!" >>>print(a.lower())
5.	replace()	replaces a string with another string	>>> a = "Hello, World!" >>>print(a.replace("H", "J"))
6.	split()	Returns a list where the text between the specified separator becomes list items	>>> a = "Hello, World!" >>>print(a.split(","))
7.	find()	finds the first occurrence of the specified value. Returns -1 if the value is not found.	>>>txt = "Hello, welcome to my world." >>>x = txt.find("e", 5, 10) >>>print(x)

8.	Escape Characters
	\' Single Quote
	\\ Backslash
	\n New Line
	\t Tab
	\\t \t
	\\n \n
	\b To remove space
		• It is used to insert special characters in string.
		• An escape character is a backslash \ followed by the character you want to insert.
		
		>>> txt = "We are called \" Indians \" by the western countries."
		Output:
		"We are called "Indians" by the western countries."
		
		>>>print('Who\'s this?')
		Output:
		Who's this?
		
		>>> print('Interview\nBit')
		Output:
		Interview
		Bit
		
		>>>print('Interview\\Bit')
		Output:
		Interview\Bit
		
		>>>print("Interview \\t Bit")
		Output:
		Interview \t Bit	

Built-in List Functions			
S. No.	**Functions**	**Description**	**Example**

1.	insert()	To insert a list item at a specified index	thislist = ["apple", "banana", "cherry"]
			
			thislist.insert(2, "watermelon")
			
			print(thislist)
			
			Output = ['apple', 'banana', 'watermelon', 'cherry']

2.	append()	To add an item to the end of the list	thislist = ["apple", "banana", "cherry"]
			thislist.append("orange")
			print(thislist)
			
			['apple', 'banana', 'cherry', 'orange']

3.	extend()	To append elements from *another list* to the current list	thislist = ["apple", "banana", "cherry"]
			tropical = ["mango", "pineapple", "papaya"]
			thislist.extend(tropical)
			print(thislist)
			
			['apple', 'banana', 'cherry', 'mango', 'pineapple', 'papaya']

4.	remove()	removes the specified item	thislist = ["apple", "banana", "cherry"]
			thislist.pop(1)
			print(thislist)
			
			['apple', 'cherry']

5.	del keyword	removes the specified index	thislist = ["apple", "banana", "cherry"]
			del thislist[0]
			print(thislist)
			['banana', 'cherry']

6.	sort()	sorts the list in ascending order by default.	thislist = ["orange", "mango", "kiwi", "pineapple", "banana"]
			thislist.sort()
			print(thislist)
			['banana', 'kiwi', 'mango', 'orange', 'pineapple']
			
			OR
			
			thislist = [100, 50, 65, 82, 23]
			thislist.sort()
			print(thislist)
			[23, 50, 65, 82, 100]

7.	sort(reverse = True)	sort the list in ascending order	thislist = ["orange", "mango", "kiwi", "pineapple", "banana"]
			thislist.sort(reverse = True)
			print(thislist)
			['pineapple', 'orange', 'mango', 'kiwi', 'banana']
			
			OR
			
			thislist = [100, 50, 65, 82, 23]
			thislist.sort(reverse = True)
			print(thislist)
			[100, 82, 65, 50, 23]

8.	reverse()	reverses the current sorting order of the elements	thislist = ["banana", "Orange", "Kiwi", "cherry"]
			thislist.reverse()
			print(thislist)
			['cherry', 'Kiwi', 'Orange', 'banana']

9.	copy()	To copy existing list	thislist = ["apple", "banana", "cherry"]
			mylist = thislist.copy()
			print(mylist)
			['apple', 'banana', 'cherry']

10.	list()	To create new list	thislist = ["apple", "banana", "cherry"]
			mylist = list(thislist)
			print(mylist)
			['apple', 'banana', 'cherry']

11.	append()	To join second list to the first	list1 = ["a", "b" , "c"]
			list2 = [1, 2, 3]
			
			for x in list2:
			  list1.append(x)
			
			print(list1)
			['a', 'b', 'c', 1, 2, 3]

12.	extend()	to add list2 at the end of list1	list1 = ["a", "b" , "c"]
			list2 = [1, 2, 3]
			
			list1.extend(list2)
			print(list1)
			['a', 'b', 'c', 1, 2, 3]

Built-in Tuple Functions

S. No.	Functions	Description	Example

1.	len()	To determine how many items a tuple has.	thistuple = ("apple", "banana", "cherry")
			print(len(thistuple))
			Output: = 3

2.	add a comma after the an item	To create a tuple with only one item	thistuple = ("apple",)
			print(type(thistuple))
			
			#NOT a tuple
			thistuple = ("apple")
			print(type(thistuple))
			Output =
			<class ='tuple'>
			<class 'str'>

Since tuples are immutable, it is first converted into the list and then append() and remove() is applied same as that of list.			

Built-in Set Functions

S. No.	Functions	Description	Example

1.	add()	To add one item to a set	thisset = {"apple", "banana", "cherry"}
			
			thisset.add("orange")
			
			print(thisset)

2.	update()	To add items from another set into the current set	thisset = {"apple", "banana", "cherry"}
			tropical = {"pineapple", "mango", "papaya"}
			
			thisset.update(tropical)
			
			print(thisset)

3.	remove()	To remove an item in a set	thisset = {"apple", "banana", "cherry"}
			
			thisset.remove("banana")
			
			print(thisset)

4.	pop()	to remove a last item	thisset = {"apple", "banana", "cherry"}
			
			x = thisset.pop()
			
			print(x)
			
			print(thisset)

5.	clear()		thisset = {"apple", "banana", "cherry"}
			
			thisset.clear()
			
			print(thisset)

6.	del keyword	To delete the set completely	thisset = {"apple", "banana", "cherry"}
			
			del thisset
			
			print(thisset)

7.	union()	returns a new set with all items from both sets	set1 = {"a", "b" , "c"}
			set2 = {1, 2, 3}
			
			set3 = set1.union(set2)
			print(set3)

8.	update()	inserts the items in set2 into set1	set1 = {"a", "b" , "c"}
			set2 = {1, 2, 3}
			
			set1.update(set2)
			print(set1)

Built-in Dictionary Functions

S. No.	Functions	Description	Example

1.	keys()	return a list of all the keys in the dictionary	thisdict = {
			  "brand": "Ford",
			  "model": "Mustang",
			  "year": 1964
			}
			x = thisdict.keys()

2.	values()	return a list of all the values in the dictionary	thisdict = {
			  "brand": "Ford",
			  "model": "Mustang",
			  "year": 1964
			}
			x = thisdict.values()

3.	items()	return each item in a dictionary, as tuples	thisdict = {
			  "brand": "Ford",
			  "model": "Mustang",
			  "year": 1964
			}
			x = thisdict.keys()

4.	update()	update the dictionary with the items from the given argument	thisdict = {
			  "brand": "Ford",

			"model": "Mustang",
			"year": 1964
			}
			thisdict.update({"year": 2020})
			print(thisdict)

5.	clear()	empties the dictionary	thisdict = {
			"brand": "Ford",
			"model": "Mustang",
			"year": 1964
			}
			thisdict.clear()
			print(thisdict)

6.	pop()	removes the item with the specified key name	thisdict = {
			"brand": "Ford",
			"model": "Mustang",
			"year": 1964
			}
			thisdict.pop("model")
			print(thisdict)

7.	del keyword	removes the item with the specified key name	thisdict = {
			"brand": "Ford",
			"model": "Mustang",
			"year": 1964
			}
			del thisdict["model"]
			print(thisdict)

8.	Adding an item	by using a new index key and assigning a value to it	thisdict = {
			"brand": "Ford",
			"model": "Mustang",
			"year": 1964
			}
			thisdict["color"] = "red"
			print(thisdict)