ESSENTIAL PYTHON METHODS EVERY DEVELOPER SHOULD KNOW

LIST METHODS

Method	Description	Code Example	Output
append()	Adds an element at the end of the list.	numbers = [1, 2, 3] numbers.append(4)	[1, 2, 3, 4]
clear()	Removes all the elements from the list.	numbers = [1, 2, 3] numbers.clear()	[ ]
copy()	Returns a shallow copy of the list.	numbers = [1, 2, 3] new_list = numbers.copy()	[1, 2, 3]
count()	Returns the number of occurrences of a specified value.	numbers = [1, 2, 2, 2, 3] print(numbers.count(2))	3
extend()	Adds elements of an iterable (like another list) to the end of the list.	numbers = [1, 2, 3] numbers.extend([4, 5])	[1, 2, 3, 4, 5]
index()	Returns the index of the first occurrence of a specified value.	numbers = [10, 20, 30, 20] print(numbers.index(20))	1
insert()	Inserts an element at a specified position.	numbers = [1, 2, 4] numbers.insert(2, 3)	[1, 2, 3, 4]
pop()	Removes and returns the element at a specified position (default is last).	numbers = [1, 2, 3, 4] numbers.pop()	[1, 2, 3]
remove()	Removes the first occurrence of a specified value.	numbers= [1, 2, 3, 2] numbers.remove(2)	[1, 3, 2]
reverse()	Reverses the order of the list in place.	numbers = [1, 2, 3] numbers.reverse()	[3, 2, 1]
sort()	Sorts the list in ascending order (or based on a custom function).	numbers = [3, 1, 4, 2] numbers.sort()	[1, 2, 3, 4]

SET METHODS

Method	Description	Code Example	Output
add()	Inserts an element into the set.	set1 = {1, 2, 3} set1.add(4)	{1, 2, 3, 4}
clear()	Removes all elements, leaving the set empty.	set1 = {1, 2, 3} set1.clear()	set()
copy()	Creates and returns a duplicate of the set.	set1 = {1, 2, 3} set2 = set1.copy()	{1, 2, 3}
difference()	Returns elements present in one set but not in another.	set1 = {1, 2, 3} set2 = {3, 4, 5} set1.difference(set2)	{1, 2}
difference_update()	Removes elements found in another set from the current set.	set1 = {1, 2, 3} set2 = {2, 3, 4} set1.difference_update(set2)	{1}
discard()	Removes a specified element if it exists; no error if absent.	set1 = {1, 2, 3} set1.discard(2)	{1, 3}
intersection()	Returns elements common to multiple sets.	set1 = {1, 2, 3} set2 = {2, 3, 4} set1.intersection(set2)	{2, 3}
intersection_update()	Updates the set with elements common to all sets.	set1 = {1, 2, 3} set2 = {2, 3, 4} set1.intersection_update(set2)	{2, 3}
isdisjoint()	Returns True if two sets have no elements in common.	set1 = {1, 2, 3} set2 = {4, 5, 6} set1.isdisjoint(set2)	True
issubset()	Checks if all elements of one set are in another.	set1 = {1, 2} set2 = {1, 2, 3, 4} set1.issubset(set2)	True
issuperset()	Checks if the current set contains all elements of another.	set1 = {1, 2, 3, 4} set2 = {1, 2} set1.issuperset(set2)	True

SET METHODS

Method	Description	Code Example	Output
pop()	Removes and returns an arbitrary element from the set.	set1 = {10, 20, 30} set1.pop())	(Random element removed, e.g.,) 10 and {20, 30}
remove()	Deletes a specified element; raises an error if not found.	set1 = {1, 2, 3} set1.remove(2)	{1, 3}
symmetric_difference()	Returns elements unique to each set (not in both).	set1 = {1, 2, 3} set2 = {3, 4, 5} set1.symmetric_difference(set2)	{1, 2, 4, 5}
symmetric_difference_update()	Updates the set with elements unique to either set.	set1 = {1, 2, 3} set2 = {3, 4, 5} set1.symmetric_difference_update(set2)	{1, 2, 4, 5}
union()	Returns all elements from both sets without duplicates.	set1 = {1, 2, 3} set2 = {3, 4, 5} set1.union(set2)	{1, 2, 3, 4, 5}
update()	Adds all elements from another set to the current one.	set1 = {1, 2, 3} set2 = {4, 5} set1.update(set2)	{1, 2, 3, 4, 5}

DICTIONARY METHODS

Method	Description	Code Example	Output
clear()	Deletes all key-value pairs, making the dictionary empty.	dict1 = {'a': 1, 'b': 2, 'c': 3} dict1.clear()	{}
copy()	Creates and returns a duplicate of the dictionary.	dict1 = {'x': 10, 'y': 20} dict2 = dict1.copy()	{'x': 10, 'y': 20}
fromkeys()	Creates a new dictionary with specified keys, assigning a common default value.	keys = ['a', 'b', 'c'] dict1 = dict.fromkeys(keys, 0) print(dict1)	{'a': 0, 'b': 0, 'c': 0}
get()	Retrieves the value associated with a key; returns None if the key is absent.	dict1 = {'a': 1, 'b': 2} print(dict1.get('b')) print(dict1.get('c'))	2 and None
items()	Returns all dictionary entries as key-value tuple pairs.	dict1 = {'a': 1, 'b': 2} dict1.items()	dict_items([('a', 1), ('b', 2)])
keys()	Provides a view containing all the keys in the dictionary.	dict1 = {'a': 1, 'b': 2, 'c': 3} print(dict1.keys())	dict_keys(['a', 'b', 'c'])
pop()	Removes a key-value pair by key and returns its value.	dict1 = {'a': 1, 'b': 2, 'c': 3} print(dict1.pop('b')) print(dict1)	2 and {'a': 1, 'c': 3}
popitem()	Removes and returns the most recently added key-value pair.	dict1 = {'a': 1, 'b': 2, 'c': 3} print(dict1.popitem()) print(dict1)	('c', 3) and {'a': 1, 'b': 2}
setdefault()	Returns the value of a key; if missing, inserts it with a specified default.	dict1 = {'a': 1, 'b': 2} print(dict1.setdefault('b', 5)) print(dict1.setdefault('c', 5))	2, 5, and {'a': 1, 'b': 2, 'c': 5}
update()	Merges another dictionary or iterable into the existing dictionary, updating keys if they exist.	dict1 = {'a': 1, 'b': 2} dict2 = {'b': 3, 'c': 4} dict1.update(dict2) print(dict1)	{'a': 1, 'b': 3, 'c': 4}
values()	Returns a view object containing all dictionary values.	dict1 = {'x': 10, 'y': 20, 'z': 30} print(dict1.values())	dict_values([10, 20, 30])

DICTIONARY METHODS

Method	Description	Code Example	Output
clear()	Deletes all key-value pairs, making the dictionary empty.	dict1 = {'a': 1, 'b': 2, 'c': 3} dict1.clear()	{}
copy()	Creates and returns a duplicate of the dictionary.	dict1 = {'x': 10, 'y': 20} dict2 = dict1.copy()	{'x': 10, 'y': 20}
fromkeys()	Creates a new dictionary with specified keys, assigning a common default value.	keys = ['a', 'b', 'c'] dict1 = dict.fromkeys(keys, 0) print(dict1)	{'a': 0, 'b': 0, 'c': 0}
get()	Retrieves the value associated with a key; returns None if the key is absent.	dict1 = {'a': 1, 'b': 2} print(dict1.get('b')) print(dict1.get('c'))	2 and None
items()	Returns all dictionary entries as key-value tuple pairs.	dict1 = {'a': 1, 'b': 2} dict1.items()	dict_items([('a', 1), ('b', 2)])
keys()	Provides a view containing all the keys in the dictionary.	dict1 = {'a': 1, 'b': 2, 'c': 3} print(dict1.keys())	dict_keys(['a', 'b', 'c'])
pop()	Removes a key-value pair by key and returns its value.	dict1 = {'a': 1, 'b': 2, 'c': 3} print(dict1.pop('b')) print(dict1)	2 and {'a': 1, 'c': 3}
popitem()	Removes and returns the most recently added key-value pair.	dict1 = {'a': 1, 'b': 2, 'c': 3} print(dict1.popitem()) print(dict1)	('c', 3) and {'a': 1, 'b': 2}
setdefault()	Returns the value of a key; if missing, inserts it with a specified default.	dict1 = {'a': 1, 'b': 2} print(dict1.setdefault('b', 5)) print(dict1.setdefault('c', 5))	2, 5, and {'a': 1, 'b': 2, 'c': 5}
update()	Merges another dictionary or iterable into the existing dictionary, updating keys if they exist.	dict1 = {'a': 1, 'b': 2} dict2 = {'b': 3, 'c': 4} dict1.update(dict2) print(dict1)	{'a': 1, 'b': 3, 'c': 4}
values()	Returns a view object containing all dictionary values.	dict1 = {'x': 10, 'y': 20, 'z': 30} print(dict1.values())	dict_values([10, 20, 30])

STRING METHODS

Method	Description	Code Example	Output
capitalize()	Converts the first letter of the string to uppercase.	str1 = "intensity coding" print(str1.capitalize())	Intensity coding
casefold()	Converts the string to lowercase for case-insensitive comparisons.	str1 = "PYTHON" print(str1.casefold())	python
center()	Aligns the string at the center with specified width.	str1 = "AI" print(str1.center(6, '*'))	**AI**
count()	Counts occurrences of a substring in the string.	str1 = "machine learning machine" print(str1.count("machine"))	2
encode()	Returns the encoded version of the string.	str1 = "AI" print(str1.encode())	b'AI'
endswith()	Checks if the string ends with a specific substring.	str1 = "deep learning" print(str1.endswith("learning"))	True
expandtabs()	Sets the tab size within the string.	str1 = "A\tB\tC" print(str1.expandtabs(4))	A B C
find()	Returns the index of the first occurrence of a substring.	str1 = "intensity coding" print(str1.find("coding"))	10
format()	Formats specified values in the string.	python str1 = "Hello, {}!".format("AI") print(str1)	Hello, AI!
format_map()	Similar to format(), but uses a mapping object.	info = {'name': 'AI'} str1 = "Hello,{name}!".format_map(info) print(str1)	Hello, AI!
index()	Returns index of substring; raises error if not found.	str1 = "machine" print(str1.index("ine"))	4
isalnum()	Checks if all characters are alphanumeric.	str1 = "AI2025" print(str1.isalnum())	True
isalpha()	Checks if all characters are alphabetic.	str1 = "AI" print(str1.isalpha())	True
isascii()	Returns True if all characters are ASCII.	str1 = "AI" print(str1.isascii())	True

STRING METHODS

Method	Description	Code Example	Output
isdecimal()	Checks if all characters are decimal digits.	str1 = "123" print(str1.isdecimal())	True
isdigit()	Checks if all characters are digits.	str1 = "12345" print(str1.isdigit())	True
isidentifier()	Validates if string is a valid Python identifier.	str1 = "variable_name" print(str1.isidentifier())	True
islower()	Checks if all letters are lowercase.	str1 = "intensity" print(str1.islower())	True
isnumeric()	Checks if all characters represent numeric values.	str1 = "2025" print(str1.isnumeric())	True
isprintable()	Returns True if all characters are printable.	str1 = "AI ML" print(str1.isprintable())	True
isspace()	Checks if the string contains only whitespace.	str1 = " " print(str1.isspace())	True
istitle()	Checks if the string follows title case.	str1 = "Machine Learning" print(str1.istitle())	True
isupper()	Checks if all letters are uppercase.	str1 = "AI" print(str1.isupper())	True
join()	Joins iterable elements using the string as separator.	str1 = "-" print(str1.join(['AI', 'ML', 'DL']))	AI-ML-DL
ljust()	Left-aligns the string in specified width.	str1 = "AI" print(str1.ljust(6, '*'))	AI****
lower()	Converts all letters to lowercase.	str1 = "INTENSITY" print(str1.lower())	intensity
lstrip()	Removes leading whitespace.	str1 = " AI" print(str1.lstrip())	'AI'
maketrans()	Generates a translation table for replacement.	trans = str.maketrans('A', 'I') str1 = "AI" print(str1.translate(trans))	II

STRING METHODS

Method	Description	Code Example	Output
partition()	Splits string into 3 parts using a separator.	str1 = "AI-ML-DL" print(str1.partition('-'))	('AI', '-', 'ML-DL')
replace()	Replaces occurrences of a substring.	str1 = "AI ML AI" print(str1.replace("AI", "DL"))	DL ML DL
rfind()	Finds last occurrence of substring.	str1 = "AI ML AI" print(str1.rfind("AI"))	6
rindex()	Returns last index of substring; raises error if not found.	str1 = "AI ML AI" print(str1.rindex("AI"))	6
rjust()	Right-aligns the string within specified width.	str1 = "AI" print(str1.rjust(6, '*'))	****AI
rpartition()	Splits string into 3 parts, searching from end.	str1 = "AI-ML-DL" print(str1.rpartition('-'))	('AI-ML', '-', 'DL')
rsplit()	Splits string from the right using a separator.	str1 = "AI,ML,DL" print(str1.rsplit(' , ', 1))	['AI,ML', 'DL']
rstrip()	Removes trailing whitespace.	str1 = "AI " print(str1.rstrip())	'AI'
split()	Splits string into a list based on separator.	str1 = "AI ML DL" print(str1.split())	['AI', 'ML', 'DL']
splitlines()	Splits string at line breaks.	str1 = "AI\nML\nDL" print(str1.splitlines())	['AI', 'ML', 'DL']
startswith()	Checks if string starts with a specific substring.	str1 = "AI ML" print(str1.startswith("AI"))	True
strip()	Removes leading and trailing whitespace.	str1 = " AI ML " print(str1.strip())	'AI ML'
swapcase()	Swaps lowercase to uppercase and vice versa.	str1 = "Ai Ml" print(str1.swapcase())	aI mL
title()	Converts first letter of each word to uppercase.	str1 = "intensity coding" print(str1.title())	Intensity Coding

STRING METHODS

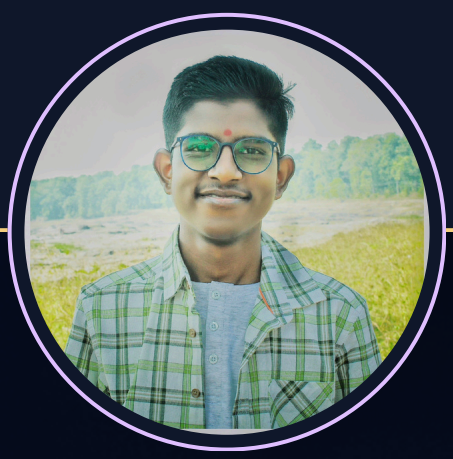
Method	Description	Code Example	Output
translate()	Modifies string using a translation table.	trans = str.maketrans({'A': '@', 'I': '1'}) str1 = "AI" print(str1.translate(trans))	@1
upper()	Converts string to uppercase.	str1 = "ai ml" print(str1.upper())	AI ML
zfill()	Pads string with leading zeros to reach desired width.	str1 = "42" print(str1.zfill(5))	00042

TUPLE METHODS

Method	Description	Code Example	Output
count()	Returns the number of times a specified value appears in a tuple.	number_tup = (1, 2, 2, 3, 2) print(number_tup.count(2))	3
index()	Searches for a specified value and returns its position (index).	number_tup = (10, 20, 30, 20) print(number_tup.index(30))	2

Found this helpful ?

Follow on LinkedIn
Master AI/ML with Intensity Coding



@bhavdippatel2020

Like | Comment | Share | Save

Each Lesson At Intensity Coding Includes Everything You Need

THEORY MADE SIMPLE
Complex ideas explained in an easy way

PYTHON CODE
Hands-on coding for practice

VISUAL LEARNING
Visual diagrams for clarity

MATH BEHIND AI/ML
Step-by-step explanation of core concepts

www.intensitycoding.com

Intensity Coding
Something For Everyone