PYTHON DATA METHODS

STRING METHODS

1. 🔠 **CASE CHANGING**

Method	Description
upper()	Converts all characters to uppercase
lower()	Converts all characters to lowercase
capitalize()	Capitalizes first letter, rest lowercase
title()	Capitalizes the first letter of every word
swapcase()	Swaps uppercase ↔ lowercase
casefold()	Aggressive lowercase (for comparisons)

2. 🧍 **ALIGNMENT & PADDING**

Method	Description
center(width, fillchar=' ')	Centers text with optional fill
ljust(width, fillchar=' ')	Left-aligns text
rjust(width, fillchar=' ')	Right-aligns text
zfill(width)	Pads with zeros on the left

3. ✂️ **REMOVING & TRIMMING**

Method	Description
strip([chars])	Removes leading and trailing chars
lstrip([chars])	Removes from left side
rstrip([chars])	Removes from right side

4. 🔍 **SEARCHING & FINDING**

Method	Description
`find(sub[, start[, end]])`	Lowest index of substring, or -1
`rfind(sub[, start[, end]])`	Highest index of substring
`index(sub[, start[, end]])`	Like `find()` but raises error if not found
`rindex(sub[, start[, end]])`	Like `rfind()` but raises error if not found
`count(sub[, start[, end]])`	Counts occurrences of substring

5. ✅ **CHECKING / TESTING**

Method	Description
`startswith(prefix[, start[, end]])`	Checks if string starts with prefix
`endswith(suffix[, start[, end]])`	Checks if string ends with suffix
`isalpha()`	True if all alphabetic
`isalnum()`	True if all alphanumeric
`isdecimal()`	True if only decimal digits
`isdigit()`	True if only digits
`isnumeric()`	True if numeric (includes fractions)
`isspace()`	True if only whitespace
`islower()`	True if all lowercase
`isupper()`	True if all uppercase
`istitle()`	True if title case
`isascii()`	True if all ASCII characters
`isidentifier()`	True if valid Python identifier
`isprintable()`	True if all printable

6. 🪓 **SPITTING & JOINING**

Method	**Description**
`split(sep=None, maxsplit=-1)`	Splits string into list
`rsplit(sep=None, maxsplit=-1)`	Splits from right
`splitlines([keepends])`	Splits by line breaks
`join(iterable)`	Joins elements using string as separator
`partition(sep)`	Splits into 3 parts: before, sep, after
`rpartition(sep)`	Like `partition` but from right

7. 🔧 **REPLACING & FORMATTING**

Method	**Description**
`replace(old, new[, count])`	Replace substring
`removeprefix(prefix)`	Removes prefix if present
`removesuffix(suffix)`	Removes suffix if present
`format(*args, **kwargs)`	String formatting
`format_map(mapping)`	Format using dictionary
`translate(table)`	Replace chars using translation table

8. 🔡 **ENCODING**

Method	**Description**
`encode(encoding='utf-8', errors='strict')`	Encodes string into bytes

SET METHODS

Method	Description
add(elem)	Adds an element to the set
remove(elem)	Removes element; KeyError if missing
discard(elem)	Removes if present; no error if missing
pop()	Removes and returns random element
clear()	Removes all elements
copy()	Returns shallow copy
union(*others)	Returns union (A ∪ B)
intersection(*others)	Returns intersection (A ∩ B)
difference(*others)	Elements in A but not B (A − B)
symmetric_difference(other)	Elements in either A or B but not both
update(*others)	Adds all elements from others (in place union)
intersection_update(*others)	Keeps only common elements
difference_update(*others)	Removes elements found in others
symmetric_difference_update(other)	Updates with symmetric difference
issubset(other)	True if A ⊆ B
issuperset(other)	True if A ⊇ B
isdisjoint(other)	True if no elements in common

LIST METHODS

Method	Description
append(x)	Adds item to end of list
extend(iterable)	Adds multiple items from iterable
insert(i, x)	Inserts item at index i
remove(x)	Removes first occurrence of x
pop([i])	Removes and returns item (default last)
clear()	Removes all items
index(x[, start[, end]])	Returns first index of x
count(x)	Counts occurrences of x
sort(key=None, reverse=False)	Sorts list in place
reverse()	Reverses order of elements
copy()	Returns shallow copy

TUPLE METHODS

Method	Description
count(x)	Returns number of occurrences of x
index(x[, start[, end]])	Returns first index of x

DICTIONARIES METHODS

Method	**Description**
clear()	Removes all key-value pairs from the dictionary (empties it).
copy()	Returns a **shallow copy** of the dictionary.
fromkeys(iterable, value=None)	Creates a new dictionary from keys in iterable, all set to the same value.
get(key[, default])	Returns the value for key if found; otherwise returns default (or None).
items()	Returns a **view object** with all (key, value) pairs.
keys()	Returns a **view object** with all keys.
values()	Returns a **view object** with all values.
pop(key[, default])	Removes and returns the value for key; if not found, returns default (or raises KeyError).
popitem()	Removes and returns **the last inserted (key, value)** pair (LIFO order since Python 3.7).
setdefault(key[, default])	Returns the value for key; if not present, inserts it with default and returns default.
update([other])	Updates the dictionary with key-value pairs from other (like merging).
__contains__(key)	Returns True if key is in the dictionary (used in key in dict).
__len__()	Returns the number of items in the dictionary (used by len(dict)).