Date: 2025 -- >July 3rd

Assignment – I
Data Cleaning Using String Functions for Variables and Data Structures

Easy

1. Remove Spaces

   Assignment: Remove leading and trailing spaces from a string using `strip()`.

   Example:

```
text = "   Hello, World!   "
cleaned = text.strip()  # "Hello, World!"
```

2. Remove Leading Characters

   Assignment: Remove all leading asterisks from a string using `lstrip()`.

   Example:

```
text = "***Welcome"
cleaned = text.lstrip("*")  # "Welcome"
```

3. Remove Trailing Characters

   Assignment: Remove all trailing exclamation marks from a string using `rstrip()`.

   Example:

```
text = "Goodbye!!!"
cleaned = text.rstrip("!")  # "Goodbye"
```

4. Capitalize a Sentence

   Assignment: Capitalize only the first letter of a lowercase sentence using `capitalize()`.

   Example:

```
sentence = "python is fun."
result = sentence.capitalize()  # "Python is fun."
```

Date: 2025 -- >July 3rd

5. Title Case a Name

   Assignment: Convert a name to title case using `title()`.

   Example:

```
name = "john doe"
result = name.title()  # "John Doe"
```

6. Clean List of Names

   Assignment: Remove leading and trailing spaces from each name in a list.

   Example:

```
names = ["  Alice", "Bob  ", "  Charlie  "]
cleaned = [n.strip() for n in names]  # ["Alice", "Bob", "Charlie"]
```

7. Remove Custom Characters

   Assignment: Remove `#` and `$` from both ends of a string using `strip()`.

   Example:

```
text = "#hello$"
cleaned = text.strip("#$")  # "hello"
```

8. Capitalize All Names in List

   Assignment: Capitalize each name in a list using `capitalize()`.

   Example:

```
names = ["alice", "bob", "charlie"]
capitalized = [n.capitalize() for n in names]  # ["Alice", "Bob", "Charlie"]
```

9. Clean Dictionary Values

   Assignment: Remove trailing spaces from all values in a dictionary.

   Example:

```
data = {"name": "Alice ", "city": "London "}
cleaned = {k: v.rstrip() for k, v in data.items()}  # {'name': 'Alice', 'city': 'London'}
```

Date: 2025 -- >July 3rd

10. Title Case Sentences in List

    Assignment: Convert each sentence in a list to title case.

    Example:

```
sentences = ["hello world", "python is fun"]
result = [s.title() for s in sentences]  # ["Hello World", "Python Is Fun"]
```

Intermediate

11. Clean and Title Case

    Assignment: Remove spaces and convert to title case.

    Example:

```
text = "   hello PYTHON world   "
result = text.strip().title()  # "Hello Python World"
```

12. Clean List of Emails

    Assignment: Remove leading/trailing spaces from each email in a list.

    Example:


```
emails = ["  alice@example.com", "bob@example.com  "]
cleaned = [e.strip() for e in emails]  # ["alice@example.com", "bob@example.com"]
```

13. Remove Leading Numbers

    Assignment: Remove all leading digits from a string using `lstrip()`.

    Example:

```
text = "12345abc"
cleaned = text.lstrip("0123456789")  # "abc"
```

14. Clean Nested List

    Assignment: Remove spaces from each string in a nested list.

    Example:

```
nested = [["  apple", "banana  "], ["  cherry  "]]
cleaned = [[item.strip() for item in sublist] for sublist in nested]  # [["apple", "banana"],
["cherry"]]
```

Date: 2025 -- >July 3rd

15. Capitalize After Cleaning

    Assignment: Clean a string and capitalize the first letter.

    Example:

```
text = "   hello world   "
result = text.strip().capitalize()  # "Hello world"
```

16. Clean Dictionary Keys

    Assignment: Remove trailing underscores from all dictionary keys.

    Example:

```
data = {"name_": "Alice", "age_": 30}
cleaned = {k.rstrip("_"): v for k, v in data.items()}  # {'name': 'Alice', 'age': 30}
```

17. Clean and Deduplicate Names

    Assignment: Clean, capitalize, and deduplicate names in a list.

    Example:

```
names = [" alice ", "Bob", "bob", "ALICE"]
cleaned = list(set(n.strip().capitalize() for n in names))  # ['Alice', 'Bob']
```

18. Remove Multiple Characters

    Assignment: Remove `*`, `-`, and spaces from both ends of a string.

    Example:

```
text = "***-hello-***"
cleaned = text.strip("*-")  # "hello"
```

19. Conditional Cleaning in List

    Assignment: Remove leading `#` only if present in each string in a list.

    Example:

```
tags = ["#python", "java", "#c++"]
cleaned = [t.lstrip("#") if t.startswith("#") else t for t in tags]  # ["python", "java", "c++"]
```

Date: 2025 -- >July 3rd

20. Clean and Group by First Letter

    Assignment: Clean and group product names by their first letter (case-insensitive).

    Example:

```
products = ["  apple", "-Banana", "apricot", "banana "]
cleaned = [p.strip(" -").capitalize() for p in products]
grouped = {}
for p in cleaned:
    key = p[^0].upper()
    grouped.setdefault(key, []).append(p)
# grouped: {'A': ['Apple', 'Apricot'], 'B': ['Banana', 'Banana']}
```

Hard

21. Clean Set of Strings

    Assignment: Clean all strings in a set of special-character-laden strings.

    Example:

```
raw = {"***Alice***", "@Bob@", "  Carol  "}
cleaned = {s.strip("*@ ").capitalize() for s in raw}  # {'Alice', 'Bob', 'Carol'}
```

22. Complex Nested Cleaning

    Assignment: Clean and title-case all strings in dictionary values (lists).

    Example:

```
data = {"fruits": ["  apple", "banana  "], "veggies": ["carrot  ", "  pea"]}
cleaned = {k: [v.strip().title() for v in vals] for k, vals in data.items()}
# {'fruits': ['Apple', 'Banana'], 'veggies': ['Carrot', 'Pea']}
```

23. Custom Title Function

    Assignment: Implement your own `title()` using only `capitalize()` and loops.

    Example:

```
def custom_title(s):
    return ' '.join([w.capitalize() for w in s.split()])
custom_title("hello world from python")  # "Hello World From Python"
```

Date: 2025 -- >July 3rd

24. Clean and Format Emails

    Assignment: Clean emails (strip spaces, lowercase all, except capitalize first letter before @).

    Example:

```
emails = ["  ALICE@Example.com", "bob@EXAMPLE.COM  "]
cleaned = [e.strip().split("@")[0].capitalize() + "@" + e.strip().split("@")[^1].lower() for e in
emails]
# ['Alice@example.com', 'Bob@example.com']
```

25. Multi-Step Cleaning

    Assignment: Remove leading numbers, trailing punctuation, and title-case the string.

    Example:

```
text = "123hello world!!!"
cleaned = text.lstrip("0123456789").rstrip("!").title()  # "Hello World"
```

26. In-Place Cleaning

    Assignment: Clean and title-case a list of strings in place (no new list).

    Example:


```
names = ["  alice ", "BOB", " charlie"]
for i in range(len(names)):
    names[i] = names[i].strip().title()
# names: ['Alice', 'Bob', 'Charlie']
```

27. Clean and Count Unique Words

    Assignment: Clean sentences, split into words, count unique words.

    Example:

```
sentences = ["  hello world  ", "Hello python "]
words = set()
for s in sentences:
    words.update(s.strip().capitalize().split())
# words: {'Hello', 'World', 'Python'}
```

Date: 2025 -- >July 3rd

28. Clean Dictionary Sentences

    Assignment: Clean and capitalize only the first word of each dictionary value.

    Example:

```
data = {"msg1": "  hello world  ", "msg2": "python is fun"}
cleaned = {k: v.strip().capitalize() for k, v in data.items()}
# {'msg1': 'Hello world', 'msg2': 'Python is fun'}
```

29. Selective Character Removal

    Assignment: Remove only leading/trailing underscores and dashes.

    Example:

```
text = "_-example-_"
cleaned = text.strip("_-")  # "example"
```

30. Batch Clean and Sort

    Assignment: Clean and title-case product codes, then sort.

    Example:

```
codes = [" code1 ", "CODE2", " code3"]
cleaned = sorted([c.strip().title() for c in codes])
# ['Code1', 'Code2', 'Code3']
```

These examples are directly runnable and demonstrate the practical use of each assignment for hands-on learning with string cleaning in Python[1][2][3].

⁂

1. https://www.geeksforgeeks.org/python/python-string-methods-set-3-strip-lstrip-rstrip-min-max-maketrans-translate-replace-expandtabs/
2. https://www.datacamp.com/tutorial/python-trim
3. https://llego.dev/posts/comprehensive-guide-common-string-methods-python/