SR UNIVERSITY

AI ASSISTED LAB EXAM-02

NAME : SHAIK FAHEEM
HTNO:2503A51L39
BATCH :20

QUESTION SUB GROUP:C

Subgroup C
C.1 — [S16C1] Debug de-duplication (case-insensitive)
Context:
Contact records in the cybersecurity SOC CRM contain duplicates differing only by case.
Your Task:

Return the first occurrence of each email (case-insensitive) while preserving original order.
Data & Edge Cases:
Input: list of emails. Normalize keys using lowercase; output retains original casing.
AI Assistance Expectation:
Use AI to spot bug (reinitializing `seen` in loop) and correct to stable algorithm.
Constraints & Notes:
Add unit tests covering mixed-case duplicates.
Sample Input
['A@x.com', 'a@x.com', 'B@y.com']
Sample Output
['A@x.com', 'B@y.com']
Acceptance Criteria: Preserves first occurrence order; case-insensitive matching

PROMPT:

Debug and fix a function that removes duplicate emails in a list. Duplicates differ only by case (e.g., A@x.com vs a@x.com). Preserve the first occurrence, keep original casing, and maintain order. Bug: seen set is reinitialized inside the loop. Correct it to a stable algorithm and add test cases.

BUGGED CODE:

```python
# ❌ Buggy version (seen resets every loop)
def buggy_dedupe(emails):
    result = []
    for email in emails:
        seen = set()  # BUG: reinitialized each iteration
        key = email.lower()
        if key not in seen:
            seen.add(key)
            result.append(email)
    return result


# --- Run buggy function with different inputs ---
print("Input:", ['A@x.com', 'a@x.com', 'B@y.com'])
print("Buggy Output:", buggy_dedupe(['A@x.com', 'a@x.com', 'B@y.com']))
print("Expected Output:", ['A@x.com', 'B@y.com'])
print()

print("Input:", ['X@z.com', 'x@z.com', 'X@Z.COM'])
print("Buggy Output:", buggy_dedupe(['X@z.com', 'x@z.com', 'X@Z.COM']))
print("Expected Output:", ['X@z.com'])
print()

print("Input:", ['one@mail.com', 'two@mail.com'])
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

BUGGED CODE OUTPUT:

```
Input: ['X@z.com', 'x@z.com', 'X@Z.COM']
Buggy Output: ['X@z.com', 'x@z.com', 'X@Z.COM']
Expected Output: ['X@z.com']

Input: ['one@mail.com', 'two@mail.com']
Buggy Output: ['one@mail.com', 'two@mail.com']
Expected Output: ['one@mail.com', 'two@mail.com']
PS C:\Users\Shaik Faheem\OneDrive\Desktop\intezam>
```

FIXED CODE :

```python
ai6.py > ...
def deduplicate_emails(emails):
    for email in emails:
        key = email.lower()  # normalize for comparison
        if key not in seen:
            seen.add(key)
            result.append(email)  # keep original form
    return result
print(deduplicate_emails(['A@x.com', 'a@x.com', 'B@y.com']))


print(deduplicate_emails(['X@z.com', 'x@z.com', 'X@Z.COM']))


print(deduplicate_emails(['a@x.com', 'B@y.com', 'A@X.com', 'b@y.com']))
#

print(deduplicate_emails([]))
```

OUT PUT OF FIXED CODE :

```
Drive/Desktop/intezam/ai6.py"
['A@x.com', 'B@y.com']
['X@z.com']
['a@x.com', 'B@y.com']
[]
PS C:\Users\Shaik Faheem\OneDrive\Desktop\intezam>
```

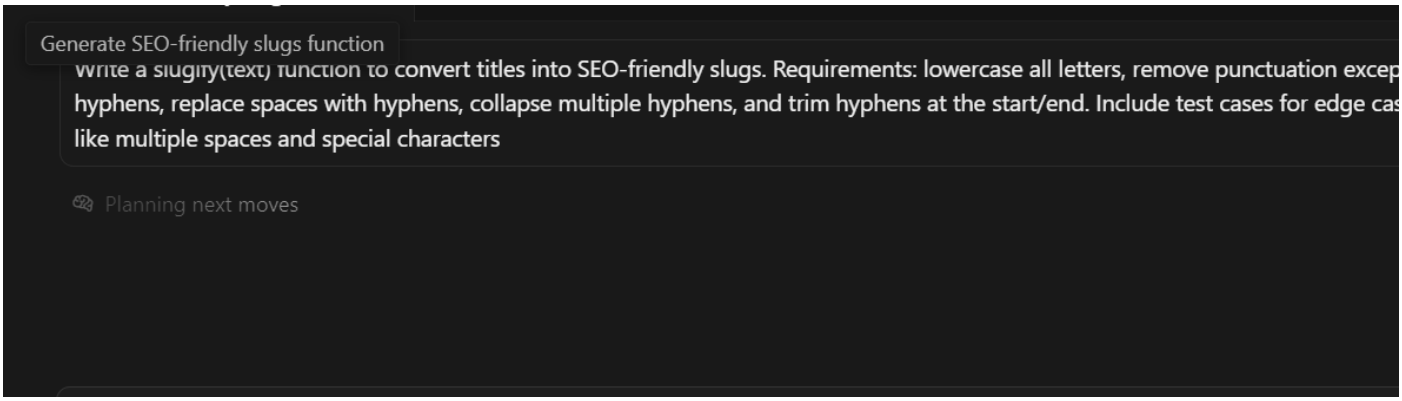
OBSERVATION: The problem is that seen is created again and again inside the loop, so the code forgets earlier emails. If we keep seen outside the loop, it will correctly remove duplicates and keep only the first email.

C.2 — [S16C2] TDD: slugify titles
Context:
Titles in the cybersecurity SOC CMS must become SEO slugs.
Your Task:
TDD for slugify(text): lowercase, remove non-alnum except hyphen, spaces->hyphen, collapse and trim hyphens.
Data & Edge Cases:
Include punctuation and multiple spaces.

AI Assistance Expectation:
Have AI propose parametric tests then implement regex solution.
Constraints & Notes:
Return correct slugs.
Sample Input
['Hello World!', 'AI & You', 'Set16-C2']
Sample Output

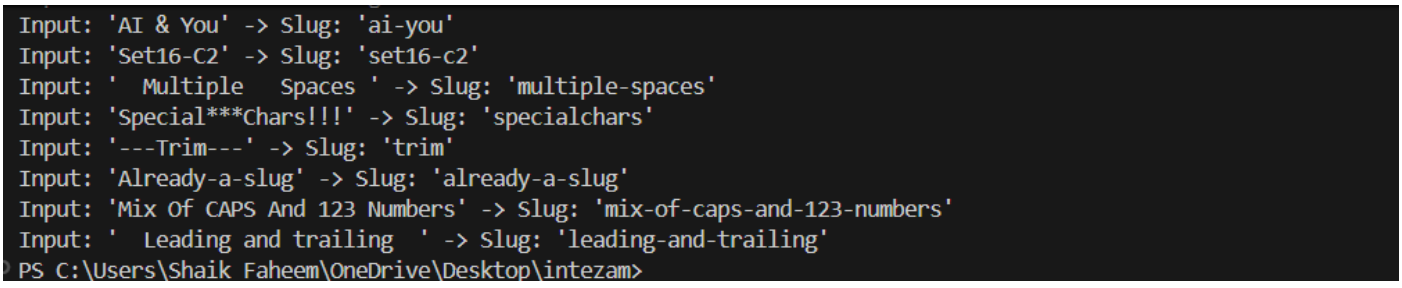
['hello-world', 'ai-you', 'set16-C2']
Acceptance Criteria: All tests pass

PROMPT:



CODE:

OUTPUT:



OBSERVATION:

The slugify function transforms titles into SEO-friendly slugs by converting all letters to lowercase, removing punctuation except hyphens, replacing spaces with hyphens, collapsing multiple hyphens into one, and trimming hyphens at the start and end.