## Subgroup C

### C.1 — [S01C1] Debug de-duplication (case-insensitive)

Scenario (e-commerce):

Context:

Customer contact lists in the e-commerce CRM contain duplicates differing only by case (e.g., 'A@x.com' vs 'a@x.com').

Your Task:

Write a function that returns the first occurrence of each email (case-insensitive) while preserving the original order.

Data & Edge Cases:

Input: list of emails. Normalize for comparison using lowercase; keep the original cased value for output.

AI Assistance Expectation:

Use AI to spot the bug (reinitializing `seen` in a loop) and propose a corrected, stable algorithm.

Constraints & Notes:

Include unit tests covering: ['A@x.com','a@x.com','B@y.com'] -> ['A@x.com','B@y.com']

Sample Input

['A@x.com', 'a@x.com', 'B@y.com']

Sample Output

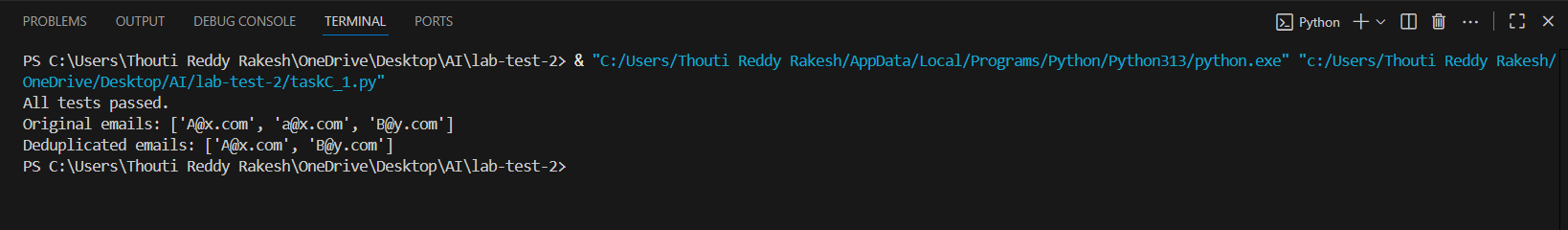
['A@x.com', 'B@y.com']

Acceptance Criteria: Preserves first occurrence order; case-insensitive matching

Code:



Output:



### C.2 — [S01C2] TDD: slugify titles

Scenario (e-commerce):

Context:

Content titles in the e-commerce CMS must become SEO-friendly slugs for URLs.

Your Task:

Design tests first for slugify(text) then implement: lowercase, remove non-alnum except hyphen, spaces->hyphen, collapse multiple hyphens, trim hyphens.

Data & Edge Cases:

Test punctuation, multiple spaces, and boundary hyphens.

AI Assistance Expectation:

Use AI to generate parameterized tests (pytest) and then implement a regex-based slugify.

Constraints & Notes:

Return correct slugs for provided samples.

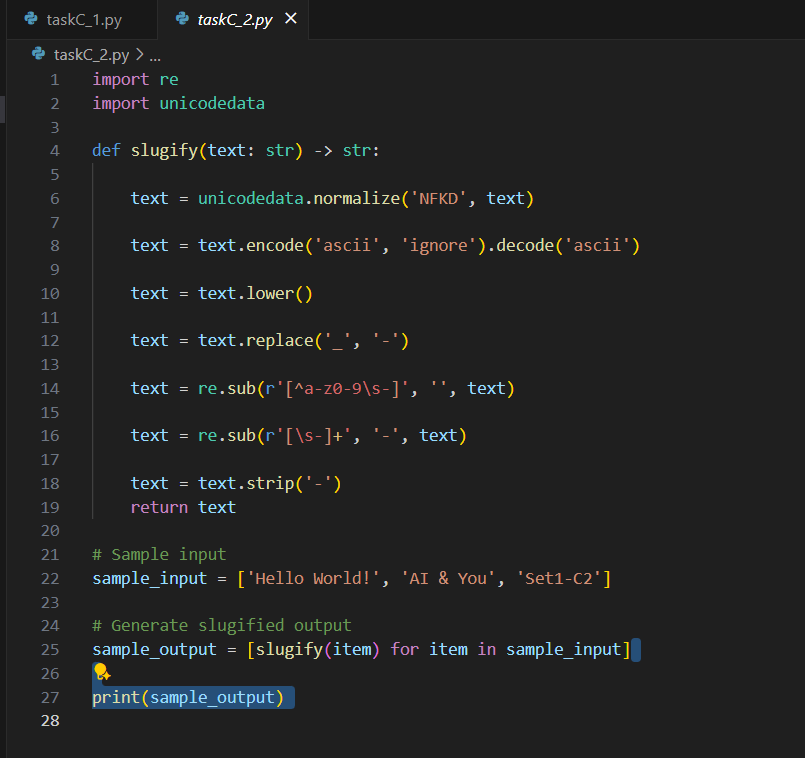
Sample Input

['Hello World!', 'AI & You', 'Set1-C2']

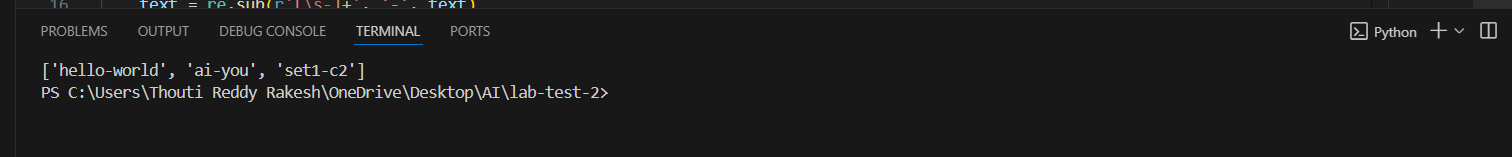
Sample Output

['hello-world', 'ai-you', 'set1-C2']

Acceptance Criteria: All tests pass; edge cases covered



Output:



------------------------------------------------------------