# LAB TEST - 2

Subgroup K

K.1 — [S13K1]

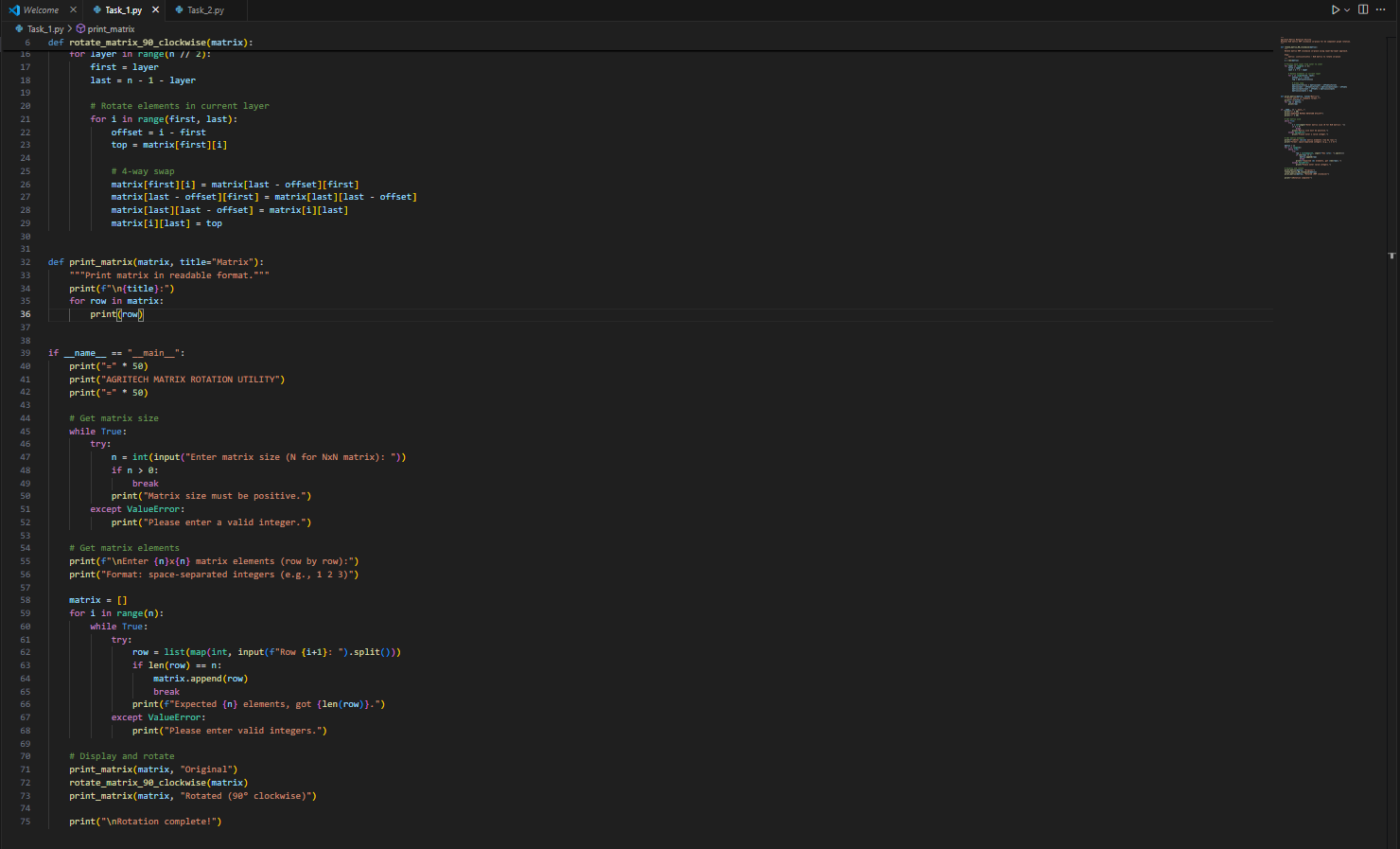
Rotate NxN matrix 90° clockwise  
Scenario (agritech):  
Context:  
A agritech UI component rotates square glyphs; engineers want an in-place matrix rotation  
utility.  
Your Task:  
Rotate an NxN matrix 90° clockwise, preferably in-place, with coverage for 1x1 and 2x2.

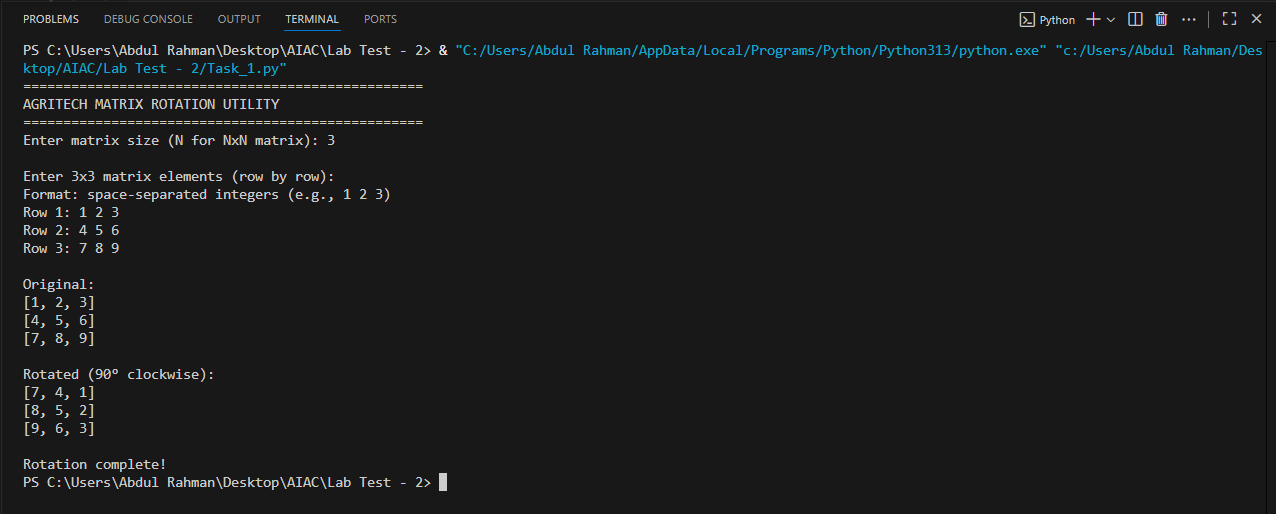
Data & Edge Cases:  
Example 3x3 shown in sample.  
AI Assistance Expectation:  
Use AI to outline layer-by-layer swaps or transpose+reverse approach; add tests.  
Constraints & Notes:  
Include tests for small N.  
Sample Input  
[[1, 2, 3], [4, 5, 6], [7, 8, 9]]  
Sample Output  
[[7, 4, 1], [8, 5, 2], [9, 6, 3]]  
Acceptance Criteria: In-place behavior correct

Prompt:

Rotate an NxN matrix 90° clockwise in-place. Use either layer-by-layer swaps or transpose + reverse method. Cover edge cases like 1x1 and 2x2 matrices; include test cases.

Screenshot:





K.2 — [S13K2] Compute added/removed lines

Scenario (agritech):  
Context:  
Change review in agritech needs a function to show added/removed lines between versions.  
Your Task:  
Given `old` and `new` lists of lines, return (added, removed) preserving the display order.  
Data & Edge Cases:  
No duplicates in outputs; do not modify input.

AI Assistance Expectation:  
Ask AI for an approach using sets but keep stable ordering via list comprehensions.  
Constraints & Notes:  
Do not show unchanged items.  
Sample Input  
old=['a','b','c'], new=['b','c','d']  
Sample Output  
added=['d'], removed=['a']  
Acceptance Criteria: Stable ordering; correct diff  
------------------------------------------------------------

Prompt:

Compare two lists of lines: old and new. Return two lists: lines added (in new but not in old ) and lines removed (in old but not in new ), preserving their order. Use sets for comparison, but list comprehensions to retain stable ordering; avoid duplicates or unchanged lines.

Screenshot :

