

AI ASSISTED CODING LAB TEST – 02

MOHAMMED ANAS SIDDIQUI

2403A52088

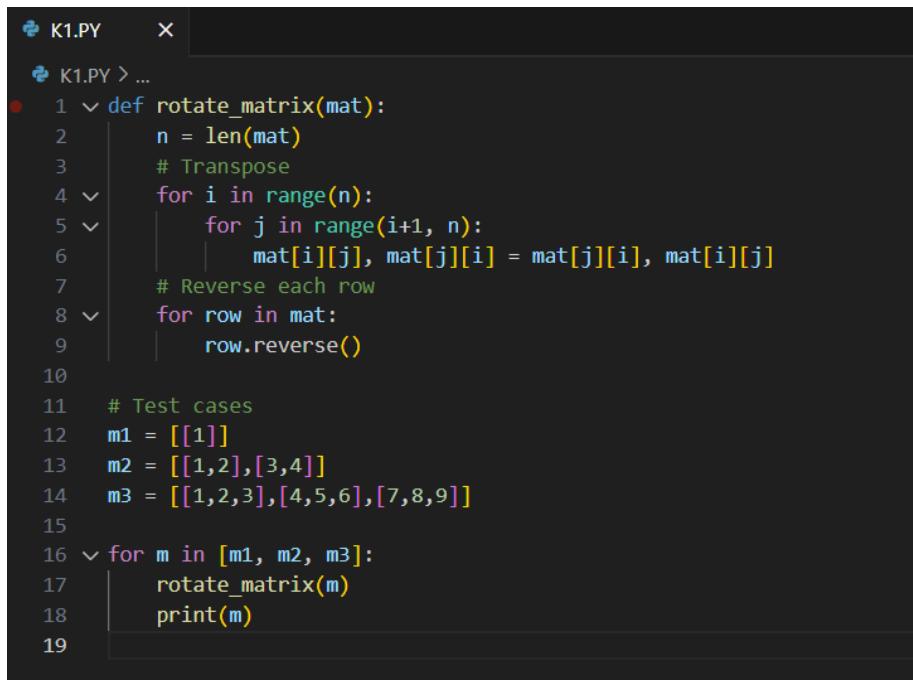
Batch – 04

K1)

Task: Rotate an NxN matrix 90° clockwise, preferably in-place, with coverage for 1x1 and 2x2

Prompt: write the code to rotate a NxN matrix, 90 degrees clockwise, preferably in place, with coverage for 1x1 and 2x2. Outline layer-by-layer swaps, or transpose + reverse approach. Add test cases and give the output.

Code:



```
  K1.PY  X
  K1.PY > ...
1  def rotate_matrix(mat):
2      n = len(mat)
3      # Transpose
4      for i in range(n):
5          for j in range(i+1, n):
6              mat[i][j], mat[j][i] = mat[j][i], mat[i][j]
7      # Reverse each row
8      for row in mat:
9          row.reverse()
10
11     # Test cases
12     m1 = [[1]]
13     m2 = [[1,2],[3,4]]
14     m3 = [[1,2,3],[4,5,6],[7,8,9]]
15
16 for m in [m1, m2, m3]:
17     rotate_matrix(m)
18     print(m)
19
```

Output:

```
PS D:\Anas\2nd Year\AIAC\Lab Test\2> & "C:/User/Test/2/K1.PY"
[[1]]
[[3, 1], [4, 2]]
[[7, 4, 1], [8, 5, 2], [9, 6, 3]]
PS D:\Anas\2nd Year\AIAC\Lab Test\2>
```

Observation: When given the prompt to copilot, it generated the code and gave the test cases for it, with correct In-place behaviour

K2)

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.

Prompt: Ask the user to give old and new lists of lines.

Return(added,removed), give the output preserving the display order.

There should be no duplicates in the output, and do not modify the input.

give the code using sets, but keep stable ordering via list comprehensions.

Do not show unchanged items.

Code:

```
❶ K2.PY > ...
1  # Function to compute added and removed lines between old and new lists
2  def diff_lines(old, new):
3      # Use sets for membership, but preserve order with list comprehensions
4      old_set = set(old)
5      new_set = set(new)
6      added = [item for item in new if item not in old_set]
7      removed = [item for item in old if item not in new_set]
8      return added, removed
9
10 # Example usage
11 if __name__ == "__main__":
12     old = ['a', 'b', 'c']
13     new = ['b', 'c', 'd']
14     added, removed = diff_lines(old, new)
15     print(f"old={old}")
16     print(f"new={new}")
17     print(f"added={added}")
18     print(f"removed={removed}")
19
```

Output:

```
PS D:\Anas\2nd Year\AIAC\Lab Test\2> & "C:/Users/Test/2/K2.PY"
old=['a', 'b', 'c']
new=['b', 'c', 'd']
added=['d']
removed=['a']
PS D:\Anas\2nd Year\AIAC\Lab Test\2>
```

Observation: When I gave the prompt to write a code to check what was added and what was removed in lists of lines, copilot swiftly gave the code, all while preserving the display order.