0x07. Rotate 2D Matrix

Algorithm Python

In a nutshell…

- **Auto QA review:** 7.15/11 mandatory
- **Altogether: 65.0%**
  - Mandatory: 65.0%
  - Optional: no optional tasks

For the "0. Rotate 2D Matrix" project, you are tasked with implementing an in-place algorithm to rotate an n x n 2D matrix by 90 degrees clockwise. This challenge requires a good understanding of matrix manipulation and in-place operations in Python. Below are the key concepts and resources that you need to grasp in order to successfully complete this project.

Concepts Needed:

1. **Matrix Representation in Python**:
   - Understanding how 2D matrices are represented using lists of lists in Python.
   - Accessing and modifying elements in a 2D matrix.
2. **In-place Operations**:
   - Performing operations on data without creating a copy of the data structure.
   - The importance of minimizing space complexity by modifying the matrix in place.
3. **Matrix Transposition**:
   - Understanding the concept of transposing a matrix (swapping rows and columns).
   - Implementing matrix transposition as a step in the rotation process.
4. **Reversing Rows in a Matrix**:
   - Manipulating rows of a matrix by reversing their order as part of the rotation process.
5. **Nested Loops**:
   - Using nested loops to iterate through 2D data structures like matrices.
   - Modifying elements within nested loops to achieve the desired rotation.

Resources:

- **Python Official Documentation**:
  - Data Structures (list comprehensions, nested list comprehension) (/rltoken/eZc_ELGxUgkuc4kkE_fd7Q)
  - More on Lists (/rltoken/0ORj179giGhGe8jpcxBkXg)
- **GeeksforGeeks Articles**:
  - In-place/Rotate square matrix by 90 degrees (/rltoken/9T8w4mtiIlRDtfLSmEmrLA)
  - Transpose a matrix in Single line in Python (/rltoken/JdIFvtej2hMW-Wd9ABHMOA)
- **TutorialsPoint**:
  - Python Lists (/rltoken/rFmzUTpaLGqDXjGA6D9eYw) for basics of list manipulation in Python.

By understanding these concepts and utilizing the provided resources, you will be able to approach the problem methodically, first transposing the matrix and then reversing each row to achieve a 90-degree clockwise rotation. This project not only tests your ability to manipulate 2D matrices but also challenges you to think about optimizing your solution to operate in-place, thus improving their problem-solving and algorithmic thinking skills in Python.

Additional Resources

- Mock Technical Interview (/rltoken/4GPWA9C2AJHtpdGxulHEPA)

Requirements

General

- Allowed editors: `vi`, `vim`, `emacs`
- All your files will be interpreted/compiled on Ubuntu 20.04 LTS using `python3` (version 3.8.10)
- All your files should end with a new line
- The first line of all your files should be exactly `#!/usr/bin/python3`
- A `README.md` file, at the root of the folder of the project, is mandatory
- Your code should use the `pycodestyle` style (version 2.8.0)
- You are not allowed to import any module
- All modules and functions must be documented
- All your files must be executable

Tasks

0. Rotate 2D Matrix

mandatory

Score: 65.0% (*Checks completed: 100.0%*)

Given an `n x n` 2D matrix, rotate it 90 degrees clockwise.

- Prototype: `def rotate_2d_matrix(matrix):`

- Do not return anything. The matrix must be edited **in-place**.
- You can assume the matrix will have 2 dimensions and will not be empty.

```
jessevhedden$ cat main_0.py
#!/usr/bin/python3
"""
Test 0x07 - Rotate 2D Matrix
"""
rotate_2d_matrix = __import__('0-rotate_2d_matrix').rotate_2d_matrix

if __name__ == "__main__":
    matrix = [[1, 2, 3],
              [4, 5, 6],
              [7, 8, 9]]
    rotate_2d_matrix(matrix)
    print(matrix)
jessevhedden$
jessevhedden$ ./main_0.py
[[7, 4, 1],
[8, 5, 2],
[9, 6, 3]]
jessevhedden$
```

Repo:

- GitHub repository: `alx-interview`
- Directory: `0x07-rotate_2d_matrix`
- File: `0-rotate_2d_matrix.py`

Check submission | >_ Get a sandbox | View results