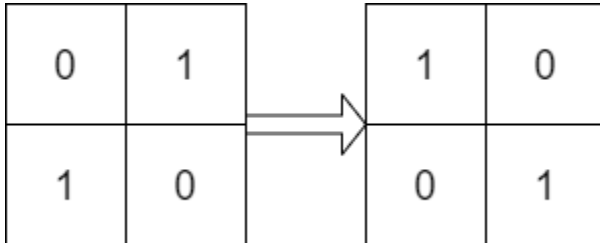


PROBLEM-1

Determine Whether Matrix Can Be Obtained By Rotation

Given two $n \times n$ binary matrices `mat` and `target`, return `true` if it is possible to make `mat` equal to `target` by **rotating** `mat` in **90-degree increments**, or `false` otherwise.

Example 1:

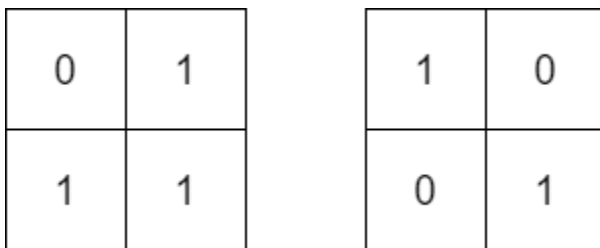


Input: `mat = [[0,1],[1,0]]`, `target = [[1,0],[0,1]]`

Output: `true`

Explanation: We can rotate `mat` 90 degrees clockwise to make `mat` equal `target`.

Example 2:

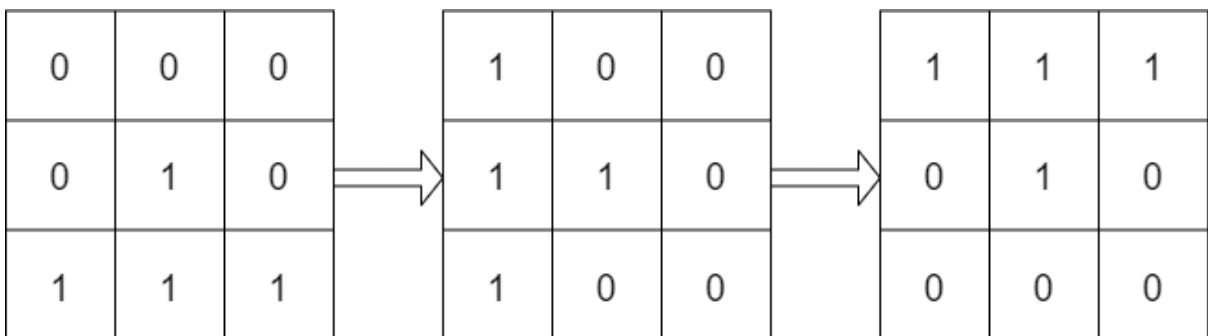


Input: `mat = [[0,1],[1,1]]`, `target = [[1,0],[0,1]]`

Output: `false`

Explanation: It is impossible to make `mat` equal to `target` by rotating `mat`.

Example 3:



Input: `mat = [[0,0,0],[0,1,0],[1,1,1]]`, `target = [[1,1,1],[0,1,0],[0,0,0]]`

Output: `true`

Explanation: We can rotate `mat` 90 degrees clockwise two times to make `mat` equal `target`.

PROBLEM-1

Constraints:

- `n == mat.length == target.length`
- `n == mat[i].length == target[i].length`
- `1 <= n <= 10`
- `mat[i][j]` and `target[i][j]` are either 0 or 1.

Solution

```
class Solution {  
  
    public boolean findRotation(int[][] mat, int[][] target) {  
  
        for (int i = 0; i < 4; i++) {  
  
            if (Arrays.deepEquals(mat, target)) {  
  
                return true;  
  
            }  
  
            rotate(mat);  
  
        }  
  
        return false;  
    }  
  
    private void rotate(int[][] mat) {  
  
        int m = mat.length;  
  
        for (int i = 0; i < m; i++) {  
  
            for (int j = i; j < m; j++) {  
  
                int temp = mat[i][j];  
  
                mat[i][j] = mat[j][i];  
  
                mat[j][i] = temp;  
  
            }  
  
        }  
  
    }  
  
}
```

PROBLEM-1

```
}

for (int i = 0; i < m; i++) {

    for (int j = 0; j < m / 2; j++) {

        int temp = mat[i][j];

        mat[i][j] = mat[i][m - 1 - j];

        mat[i][m - 1 - j] = temp;

    }

}

}
```

OUTPUT

The screenshot shows a code editor with the following C++ code:

```
23 for (int j = 0; j < m / 2; j++) {
24     int temp = mat[i][j];
25     mat[i][j] = mat[i][m - 1 - j];
26     mat[i][m - 1 - j] = temp;
27 }
28 }
29 }
30 }
```

Below the code editor, the 'Run Code Result' tab is active, showing the following details:

- Accepted** Runtime: 0 ms
- Your input:** `[[0,1],[1,0]]`
- Output:** `true`
- Expected:** `true`

At the bottom of the interface, there are buttons for 'Next >', 'Console', 'Use Example Testcases', 'Run Code', and 'Submit'. The system tray at the very bottom shows the time as 20:15 on 20-06-2021.